FACE FOUNDER BY ALMOG GAL

מבנה הספר

מבוא

תיק הפרויקט הזה יעבור על כלל הנושאים המקיפים בהם עוסק הפרויקט, יעבור על דרך ביצוע אלגוריתמים מסיימים, יעבור על פרוטוקול השרת והלקוח, יעבור על למה בחרתי את הפרויקט, לאיזה מטרה ובעיה הפרויקט בא לפתור, סדר התיקיות והצורה שבא הפרויקט בנוי, קבצים מסיימים בפרויקט, תהליך מחקר ורקע לפרויקט, קשיים ופתרונותיהם, רפלקציה
אישית של, אופן הכנת מסמכים מלווים לפרויקט, הגבלות ובעיות הפרויקט ובבלוגרפיה.

כאשר בחרתי לעשות את הפרויקט הייתי צריך לבחור באיזשהו נושא ורעיון שעליו בעצם יהיה בפרויקט. בעודי חושב, הסתכלתי על החדשות וראיתי תצלום של מצלמות שמזהות אנשים מסיימים ודעים מי הם. מאוד התלהבתי מהעניין ולכן בחרתי לקרוא על כך
באינטרנט. די הנחתי לא משנה מה הרעיון או הנושא שלי יהיה יש נושא ליבה שעליהם גם ככה אצטרך ללמוד כגון: בסיס נתונים, שרת לקוח, ממשק משתמש גרפי וכדומה. לכן בנושאים אלו פחות התמקדתי כי בכול מקרה הייתי לומד אותם לא משנה מה היה הנושא שלי. התמקדתי רק על רעיון הזיהוי פנים וזיהוי מי הוא האדם. ראיתי כמה דרגות קושי בין
יצירת הכל מהתחלה בעזרת מכונה לומדת, בעצם ללמד מכונה לזהות ולהשוות בין פרצופים, לבין להכין קוד בעזרת מכונה לומדת שכבר אומנה. החלטתי שהחלטת דרגת הקושי בנושא הזה תהיה בסוף הפרויקט בהתאם לזמן ולכוח שישאר לי. אבל כן, הנושא סיקרן אותי
והתחלתי לקרוא ולראות סרטונים בנושא מכונה לומדת. בסופו של יום, החלטתי שעל זה
יהיה הפרויקט שלי.

תחילה, ביצעתי סקר שוק. כדי לראות האם יש אפליקציות שמבצעות את מה שאני עושה. אם לא, אומנם הפרויקט יהיה קשה אבל מיוחד וטוב. אם כן, שזה היה המקרה, אז אוכל להיעזר באלגוריתמים של אותה אפליקציה. במהרה מצאתי שישנה אפליקציה די מרכזית
בחיים שמבצעת זיהוי פנים וההשוואה, רק משתמשת בה לדרך קצת שונה וזהו כמובן גוגל
תמונות. כל תמונה שאנו מצלמים בטלפונים החכמים שלנו עולה די אוטומטית לגוגל תמונות, גוגל תמונות מאתר את הפרצופים בתמונה, משווה ביניהם לבין פרצופים שכבר צלמו, ויצר לך תיקיות של כל הפרצופים שכבר צלמו. לכן הסתכלתי ומצאתי את המכונה הלומדת שבא
השתמשה גוגל ותחלה השתמשתי בה. האומנם הפרויקט שלי מסתכל על דברים אחרת,
שכן, אני מאתר אנשים מסיימים בתמונות בעוד שגוגל מסדרת תמונות של אנשים בתיקיות. שכן אלגוריתם כמו שלי יש במצלמות אבטחה כדי לאתר פושעים למשל.
אז הפרויקט שלי די מחדש את השוק כי התקשיתי למצוא איזשהי אפליקציה בשוק שעושה בדיוק מה
שהפרויקט שלי עושה.

התמודדתי עם הרבה בעיות במהלך הפרויקט שכן עד תחילת עשיית הפרויקט לא הצלחתי אפילו לכתוב שרת ולקוח. מנגד, החלטתי כן לנסות ולהשקיע המון זמן, בניגוד לקבוצה של תלמידים שפרשו, לאחר למידת הבסיס, טוב ככל שיכלתי, התחלתי לבצע את הפרויקט. היו לי 2 קשיים מרכזים בכתיבת הפרויקט: ממשק משתמש גרפי- שימוש בtkinter הקשה עלי מאוד — אבל אם השקעת המון זמן הבנתי את העקרונות והסתדרתי. מכונה לומדת- תחילה הרעיון היה ליצור ולאמן אחת כזאת, אם כי ככל שקראתי על הנושא הבנתי כמה קשה זה
יהיה בשבילי והחלטתי להשאיר את זה לסוף.

המוטיבציה שלי להמשיך את הפרויקט למרות כל הקשיים היתה מכמה סיבות: אחד נהניתי בהכנת פרויקט באסמבלי בכיתה י' ולכן הנחתי שגם הפרויקט הזה יהיה מהנה, רצון לעוד 5 יחידות בתעודת הבגרות מה שיעזור לי בהמשך דרכי, ניסיון ידע שאצבור עזור לי גם מבחינת פיתוח המוח וגם מבחינת ידע במחשבים, ידע זה יתרום לי לקבלת תפקיד בצבא
שקשור למחשבים ובכן יתרום לי לעבוד במקצוע הזה באזרחות דבר שאני מאוד מעוניין בו והסיבה האחרונה שאני מאוד אוהב לאתגר את עצמי גם באתגרים שנראים לי קשים ובלתי
אפשריים כדי להראות לעצמי שאני יכול.

הצורך הראשוני שהפרויקט ענה עליו הוא מציאת איזשהו אדם בין מקבץ תמונות. ז"א יש לפרויקט המון שימושים למשל צלם איחעים יכול להעלות את כל התמונות שצילם וחפש

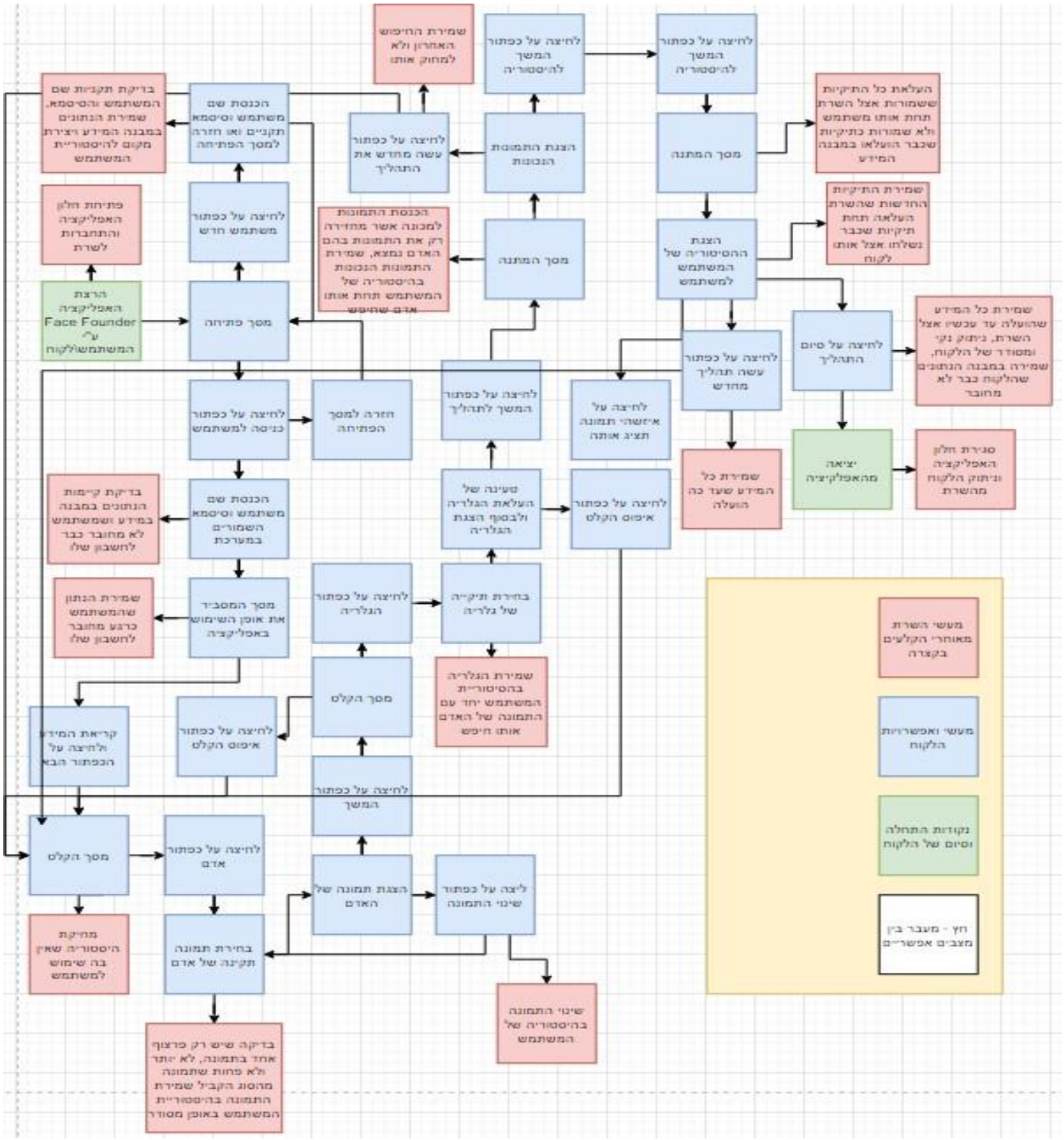
אלמוג גל. Face Founder.

בשביל אנשים תמונות שבהם הם מופיעים)כמובן, שהאפליקציה שומרת את ה'סטוריית
המשתמש(או שימוש אבטחתי כמו למצוא אדם חשוד בין מקבץ תמונות של חשודים. הפתרון הוא מציאת האנשים, תוך כדי שמירת ה'סטוריה של
המשתמש באופן מאובטח
ומסודר.

מבנה / ארכיטקטורה של הפרויקט

הצגת הפתרון המוצע והסיבות לבחירתו מוצגות במבוא)עמוד 3, פסקה 2 ו6(.

Top-Down Level Design של ארכיטקטורת הפרויקט:



*** כמובן שבכל חלון המשתמש יכול לצאת מן האפליקציה, ולהתנתק מן השרת בצורה לא מסודרת אך נקייה. השרת ישמור במנה הנתונים שהמשתמש כבר לא מחובר לחשבון שלו.**

תפקידי יחידות הפרוייקט:

תרחישים עיקריים: בפרויקט ישנם תרחישים עיקרים המחולקים לחלונות שלכל חלון יש מטרה מסוימת, ודרכים להגיע לחלונות ספציפיים אחרים. בפרויקט עתיד להיות ממשק משתמש גרפית באמצעות tkinter. התרחישים העיקרים על פי החלונות בפרויקט הם:

הרשמה וכניסה- לאחר הרצת קובץ exe ע"י המשתמש יפתח לו מסך הבית ובו 3 כפתורים: "משתמש חדש", "כניסה".

לחיצה על לחצן "משתמש חדש" תפתח בפני המשתמש חלון בו 2 תאי מילוי: משתמש וסיסמא. על שם המשתמש וסיסמא להיות באורך לפחות 4 תווים ועל שם המשתמש לא להיות כבר קיים במערכת. לאחר הרשמה תקינה יחזור המשתמש למסך הבית בו יהיה לו רשום "ההרשמה בוצעה בהצלחה. שלום ותודה שהשתמשת ב'זיהוי פנים'." השרת יבדוק את תקינות הקלט ואחר מכן ישמור את הנתונים התקנים שהמשתמש הכניס בבסיס הנתונים שלו.

לחיצה על כפתור "כניסה" תפתח בפני המשתמש 2 תאי מילוי: משתמש וסיסמא. על שם המשתמש להיות קשור אל הסיסמא ועל שם המשתמש והסיסמא להיות שמורים במערכת. לאחר כניסה תקינה יועבר המשתמש למסך ההוראות. השרת יבדוק שאכן שם המשתמש והסיסמא קשורים זה לזה, שמורים, ושקלט תקין בעזרת בסיס הנתונים. לאחר מכן ישנה בבסיס הנתונים את מצב הסטטוס של הלקוח למחובר.

בכל חלון בסשן זה יהיה כפתור "חזור" שבאמצעותו המשתמש יוכל לחזור למסך הבית המערכת תקפיץ למשתמש הודעה שגיאה בכל פעם שקורה אחת.

הוראות- לאחר ההתחברות של המשתמש אל החשבון שלו יופיע לו מסך הוראות וכפתור "הבא".

במסך ההוראות יהיה רשום למשתמש איך להשתמש באפליקציה:"1.העלה תמונה של אדם – תן למערכת Pathלתמונה של אדם מסוים. השתדל שהתמונה תהיה ברורה ככל שאפשר. אפשר להעלות רק אדם אחד בכל פעם. 2.תעלה לאפליקציה תמונות ע"י בחירת תיקיית גלריות מסוימת 3. אם ישנם תמונות אשר הפנים של האנשים בה אינם ברורים סיסי" התמונה למצוא את האדם הרצוי פוחתים. 4. בסוף התהליך התמונות שימצאו לנכון להיות התמונות בהם האדם נמצא ישמרו באופן אוטומטי במשתמש שלך. בנוסף, תהיינה לך גישה להיסטורי שלך דרך כניסה למשתמש שלך באפליקציה".

לחיצה על כפתור "הבא" תעביר את המשתמש אל מסך קלט.

בנוסף יהיה כפתור חזור שלחיצה עליו תחזיר את המשתמש אל מסך הבית. **קלט-** מטרת המסך היא לקבל את המידע הנחוץ לאפליקציה על מנת לפעול- קלט מן המשתמש. בפני המשתמש 3 כפתורים: "אדם", "גלריית תמונות" ו"איפוס הקלט". כמו כן יהיה כפתור "חזור" שיחזיר את המשתמש למסך ה"הוראות". השרת פה "מאחורי הקלעים" מוחק את כל ההיסטוריה הלא נחוצה של המשתמש".

לחיצה על כפתור "אדם" תוביל את המשתמש אל חלון שבו יבחר את האדאם ברצונו לחפש. בנוסף על המערכת לבדוק האם הPath קיים, האם כל הקבצים בו מסוג jpg או png, האם המשתמש העלה תמונה בה אין אדם או יותר מאדם אחד. המערכת תודיע למשתמש על כל טעות אפשרית. לאחר הכנסת Path תקין יעבור למסך "אמצע הקלט", בוא תהיה לו האפשרות או לעשות תהליך הקלט שוב ולמחוק את הנוכחי או להמשיך להעלאת הגלריה. השרת יבדוק את תקינות התמונה ישמור את התמונה תקינה תחת ההיסטוריה של אותו משתמש.

לחיצה על כפתור "גלריית תמונות" תוביל את המשתמש לחלון בו המשתמש יבחר איזשהי תיקיית תמונות בה הוא רוצה לחפש את האדם. פה לא תהיינה הגבלה של כמות או של סוג

אלמוג גל. Face Founder.

הקובץ אלא המערכת תודיע מראש שקובץ שאינו מסוג jpg או png, המערכת תבדוק האם הPath לגלריית התמונות קיים. השרת יבדוק את תקינות הקלט ושמור את הגלריה
בהיסטוריה של אותו משתמש תחת אותו אדם.

לאחר העלאת הגלריה המשתמש יעבור למסך "סיום הקלט", בוא תהיה לו האפשרות או לעשות תהליך הקלט שוב ולמחוק את הנוכחי או להמשיך לחלון ההמתנה. "המתנה לקלט"
בו יחכה יהיה רשום "חכה בבקשה שהמערכת תסיים לבדוק את תקינות התמונות...".

לחיצה על כפתור "איפוס הקלט" תאפס את כל הקלט ששל המשתמש בתהליך הספציפי הזה ותמחק את התהליך הנ"ל מההיסטוריה של השרת.
המשתמש יחזור מחדש למסך
ה"קלט".

לאחר הגסת כל הנתונים יהיה למשתמש כפתור "המשך לתהליך" שלחיצה עליו תוביל
למסך "המתנה לסיום תהליך הזיהוי".

המתנה לסיום תהליך הזיהוי- מסך בו יהיה איזשהו עם מדד גרפי בו יהיה רשום כמה תמונות נבדקו עד כה מתוך כמה תמונות, עם הכיתוב "חכה בבקשה שהמערכת תסיים את התהליך..." המשתמש יהיה תקוע במסך זה עד שהמערכת תסיים את התהליך. כאשר התהליך סיים המשתמש יעבור למסך ה"בדיקה וסיום התהליך". השרת יקח את תמונות
הגלריה ואת תמונת האדם ובעזרת מכונה לומדת ימצא באילו תמונות האדם אכן נמצא. לאחר מכן ישמור השרת את התמונות הנכונות ההיסטוריה של אותו משתמש תחת אותו
אדם.

בדיקה וסיום התהליך- במסך זה יהו 2 כפתורים: "עשיית התהליך שוב" ו"לעבור
להיסטוריה" ותמונת האדם\אנשים שהמערכת מצאה.

לחיצה על כפתור "עשה את התהליך שוב" תחזיר את המשתמש למסך הקלט הריק ותשמור את החיפוש האחרון שלו בהיסטוריה, אך לא תעלה אותו כי המשתמש לא הגיע למסך ההיסטוריה. לחיצה כל כפתור "לעבור להיסטוריה" תעביר את המשתמש ל"מסך ההמתנה
להעלאת ההיסטוריה".

מסך ההמתנה להעלאת ההיסטוריה- העלאת ההיסטוריה ללקוח ושמירתה במחשב תחת
הpath של האפליקציות במחשבי windows. השרת יבצע בדיקה בבסיס הנתונים אילו תיקיות היסטוריה השרת כבר העלה, וכן מתוך כל התיקיות שקיימות אצל השרת, הוא ידע אילו תיקיות הוא צריך לעלות. השרת יעלה את התיקיות הרלוונטיות והחדשות ללקוח. לאחר מכן ישמור\יוסף לבסיס הנתונים תיקיות שכבר הוסף. כך ידע בפעם הבאה מה לעלות, ובכך השרת מייעל את עבודתו ובאופן כללי את הקוד. לאחר סיום הלקוח יעבור לחלון ה"היסטוריה
האישית".

היסטוריה אישית- במסך זה יהיה מוצג במעין תיקיות ששמם שם האדם שחיפשת עליו ובכל תיקייה את התמונות שבהם האדם נמצא ותצג תמונת האדם\אנשים שהמערכת חיפשה)בצד בקטן(. המשתמש יכל לעיין בהיסטוריה האישית שלו כרצונו ואף להסיר תמונות. במסך זה יהיה כפתור "קלט נוסף" אשר לחיצה עליו תוביל את המשתמש אל מסך ה"קלט". וכפתור
"סיום תהליך ויציאה" אשר יציא יציאה מסודרת המשתמש מן האפליקציה.

*כל התנתקות תמחק את כל המידע שעוד לא נשמר במערכת ואת כל הקלטים עד כה.

גבולות הפרוייקט- לפרויקט ישנם את הגבולות הבאים:

הגבלת סוג המידע- הפרויקט יכל למצוא פנים רק בקבצים מסוגים ספציפיים: jpg, png .
ועל כן ישנה הגבלה מבחינת המשתמש שצריך לדאוג שהמידע שהוא מעביר לאפליקציה
תואם בסוגו.

הגבלות כוח זיכרון- מכיון שתיכון הdata base של צורך הרבה זיכרון, יכל להיווצר מצב
של עודף מידע בdata base, כמובן מבחינה תאורטית אם לאפליקציה תהיה רב משתמשים.

הגבלות כוח מעבד- האפליקציה תפעל עם threads כך שתוכל לבצע כמה פעולות במקביל, ובכל לטפל בכמה לקוחות במקביל ולא תיתקע בכל בקשה שדורשת כוח מעבד ובכך זמן ריצה רב. ולכן, שוב מבחינה תאורטית, האפליקציה תפעל לאט יחסית משימוש רב משתמשי כי היא מוגבל בכוח מעבד.

הגבלות מהירות- יתכן שבגלל הסיבות שציינת לגבי כוח הזיכרון והמעבד יחס המשתמשים זמן מה בין חלונות, ולכן יש לי גיבוי בחלון ה"המתנה".

הגבלת תוכנה – האפליקציה תעבוד רק למשתמשי windows.

מבני נתונים שנעשה בהם שימוש:

בפרויקט, בעיקר מעניין נוחות השתמשתי ברשימות, גם כי אני כותב בפייתון, שבו רשימות הם הדבר היותר מקובל)בניגוד לשפות c למשל(. בנוסף גם במבנה -nיה סדורה)tuple(. אם כן, ישנם פעמים שהשתמשתי ברשימה שבעצם דימתה מבנים אחרים כמו למשל רשימה שדימתה מחסנית. דוגמא לשימוש רגיל של רשימה הוא לשמור ברשימה את כתובת הלקוח יחד עם מפתח ההצפנה שלו , בnיה סדורה, כל חיבור של לקוח חדש. כך השרת ידע באיזה אופן להצפין לכל לקוח. דוגמא לשימוש של רשימה שדימתה מחסנית היא כאשר לקחתי כל התיקיות של אותו לקוח תחת אדם מסוים, מכיוון שהפקודה של הוצאת תיקיות מpath מסוים מסדרת את התיקיות לפי זמנים, בעצם כמו במחסנית, האחרון שהוכנס הוא במעלה המחסנית)LIFO(, יכולתי לדעת שהתיקיה הראשונה ברשימה תהיה התיקיה עם התמונות הנכונות של אותו אדם, כאמור תמיד תיקיה זו תיווצר אחרונה.

זרימת המידע בין היחידות השונות:

בעצם, נקטתי בשיטה של סדר שנשמרת לאורך כל ה"מסע" של הלקוח. כל נתון רלוונטי של הלקוח ישר נשמר אצל השרת. כך לא יאבד מידע. ולאחר מכן, כל מה שנשאר זה תיאום, בעצם, שהשרת תמיד "ידע" איפה ב"מסע" כל לקוח נמצא. הסדר ד' כחנולוגי בפרויקט, חוץ מכמה לולאות דבר שמאפשר לשרת לדעת שהמידע תמיד באיזה סדר מסוים אותו הוא יודע, ומפשט דברים ללקוח.

ארכיטקטורת הרשת:

פרוטוקולי הרשת: ההודעות ברשת מועברות באמצעות מערכים, למען נוחות. לכן כל הודעה עוברת pickle ומתורגמת בעצם מרשימה להודעת string וכמובן בצד השני גם להפך)loads\dumps(.

הודעות הלקוח לשרת:

הסבר	שם משתמש\ ממי נשלחה ההודעה	מידע נחוץ נוסף	מידע נחוץ	הבקשה\הפעולה	סוג ההודעה
בקשה ליצור משתמש חדש עם הסיסמא ה1234 מאת המשתמש Almog	Almog		1234	CNU (create new user)	re (request)

re (request)	LTA (login to account)	1234		Yossi	בקשה להיכנס אם הסיסמא 1234 אם המשתמש של Yossi
ok (waiting)		NP (no password)		Almog	הודעת שנועדה שהלקוח Almog שלח הודעה כדי שיהיה מוכן לקבל הודעה מהשרת, ולהודיע לשרת על תפקוד הלקוח Almog
re (request)	exit	NP (no password)		Almog	הודעה של הלקוח על יציאת המשתמש Almog מהאפליקציה
re (request)	IPI (insert person image)	מידע של התמונה מקריאה בינארית שלה	Trump.png	Almog	בקשת הכנסת תמונה למשתמש Almog של אדם-Trump אותו יחפשו, מידע של התמונה, באמצעותו השרת ישמור אצלו את התמונה, ושם התמונה-Trump.png
re (request)	IGF (insert gallery folder)	My Gallery	Almog		בקשת יצירת גלריה בשם My Gallery אצל הלקוח Almog

re (request)	IPTG (insert person to gallery)	מידע של התמונה מקריאה בינארית שלה	Moshe.jpg	Almog	בקשת הכנסת תמונה של Moshe תחת השם Moshe.jpg לגלריה האחרונה העכשווית אצל הלקוח Almog השרת בודק גם את תקינות התמונה, סוג ושיש בה פרצוף אחד.
re (request)	FTP (find the person)	NP (no password)		Almog	בקשת מציאת האדם העכשווי בגלריית התמונות של האדם הנ"ל של הלקוח Almog
re (request)	SNHF (send new history folders)	NP (no password)		Almog	בקשה לשלוח תיקיות היסטוריה שעוד לא שלחת ללקוח Almog
re (request)	SITH (send images to history)	NP (no password)		Almog	בקשה לשלוח תמונות לתיקיות היסטוריה החדשות ,תוך כדי שתשלח לי איזה תמונה לאיזה תיקייה, ללקוח Almog

בקשה לשליחת מפתח ההצפנה הגלוי	NU (no username)		NP (no password)	GPK (get public key)	re (request)
בקשה שהשרת יצור מפתח סודי ואישי ללקוח שממנו נשלחה ההודעה בעזרת המספר 14641 שנוצר. בעזרת RSA	NU (no username)	14641	NP (no password)	CASK (create a symmetrical key)	re (request)

* כמובן כאשר אין מידע נחוץ נוסף הפאקטה בגודל 4 ואין בה תא ריק.

* שליחת ראשי התיבות של ההודעות כמובן, מוסכמת על השרת והלקוח, נועדה לחסוך גודל של מידע ועומס מיותר על השרת וגם נועדה ליצור עד איזשהי רשת ביטחון כאשר יהיו שההודעה בין 3 ל4 תווים.

* ברוב המקרים השרת מחזיר תשובה בהתאם להודעה של הלקוח.

הודעות השרת ללקוח:

הסבר	מי נשלחה ההודעה	מידע נחוץ נוסף	מידע נחוץ	הפעולה שבוצעה	סוג ההודעה
השרת אישר את יציאת הלקוח – יציאה נקייה ומסודרת	Server			exit	CO (confirmed)
שליחת המפתח הגלוי	Server	145678	5	PKS (public key sent)	CO (confirmed)

CO (confirmed)	**SKS (symmetrical key saved)**	**5**	**145678**	**Server**	**הודעה שהמפתח הסימטרי סודי אישי ללקוח נשמר בהצלחה. בנוסף מצורף המפתח הציבורי להודעה.**
ER (error)	**NU (no username)**			**Server**	**השרת מודיע על שגיאה בשל אי הכנסת שם משתמש בעת יצירת משתמש חדש**
ER (error)	NP (no password)			**Server**	**השרת מודיע על שגיאה בשל אי הכנסת סיסמא בעת יצירת משתמש חדש**
ER (error)	**UTS (username too short)**			**Server**	**השרת מודיע על שגיאה בשל הכנסת שם משתמש קצר מדי בעת יצירת משתמש חדש**
ER (error)	**PTS (password too short)**			**Server**	**השרת מודיע על שגיאה בשל הכנסת סיסמא קצרה מדי בעת יצירת משתמש חדש**

ER (error)	UAE (username already exists)			Server	השרת מודיע על שגיאה בשל הכנסת שם משתמש שכבר קיים בעת יצירת משתמש חדש
CO (confirmed)	NUC (new user created)			Server	השרת מודיע ללקוח שהחשבון נוצר בהצלחה
ER (error)	UOPW (user or password are wrong)			Server	השרת מודיע ללקוח על שגיאה בשל הכנסת שם משתמש או\גם סיסמא שגויה
ER (error)	AAIU (account already in use)			Server	השרת מודיע ללקוח על שגיאה בשל ניסיון להיכנס לאותו חשבון במקביל
CO (confirmed)	PGI (person got inserted)			Server	השרת מודיע ללקוח שהתמונה של האדם שהעלה נשמרה ושנפתחה לו תיקיית היסטוריה בהצלחה

CO (confirmed)	**GGC (gallery got created)**			**Server**	**השרת מודיע ללקוח שיצירת תיקיית הגלריה לאדם אותו הוא מחפש נוצרה בהצלחה**
CO (confirmed)	**PITG (people inserted to gallery)**			**Server**	**השרת מודיע ללקוח שהעלאת הגלריה, של האדם אותו הוא מחפש, בוצעה בהצלחה**
CO (confirmed)	**PITI (person in this image)**	**Trump.jpg**	מדע של התמונה מקריאה בינארית שלה	**Server**	**השרת מודיע ללקוח שהאדם שחיפש נמצא בתמונה Trump.jpg מתוך הגלריה שהלקוח העלה לאותו אדם. האדם הוא האדם העכשווי, האחרון שהועלה.**
CO (confirmed)	**NMIF (no more imaged found)**			**Server**	**השרת מודיע ללקוח שלא נמצאו יותר תמונות בהם מופיע האדם אותו הלקוח חיפש בגלריה שאותו צירף לאותו אדם**

CO (confirmed)	**HFS (history folders send)**			**Server**	**השרת מודיע ללקוח ששלח את תיקיות ההיסטוריה החדשות, שהשרת לא העלה לפני.**
CO (confirmed)	**IITH (insert image to history)**	**Trump**	**Person** **two more places in array: Trump.jpg** מדע של התמונה מקריאה בינארית שלה	**Server**	**השרת מודיע ללקוח שישנה עוד תמונה שיש להעלות להיסטוריה של אותו קוח. עליו להעלות בתיקייה של Trump בתת התיקייה Person את התמונה Trump.jpg וזאת יעשה באמצעות המידע של התמונה**
CO (confirmed)	**NMHF (no more history files)**			**Server**	**השרת מודיע ללקוח שסוימה ההעלאה של ההיסטוריה, אין יותר**

קבצי היסטוריה חדשים					

***** כמובן כאשר אין מידע נחוץ נוסף הפאקטה בגודל 4 ואין בה תא ריק.

* שליחת ראשי התיבות של ההודעות כמובן, מוסכמת על השרת והלקוח, נועדה לחסוך גודל של מידע ועומס מיותר על השרת וגם נועדה ליצור עוד איזשהי רשת ביטחון כאשר יהיע
שההודעה בין 3 ל4 תווים.

* ברוב המקרים השרת מחזיר תשובה בהתאם להודעה של הלקוח. * השרת שולח את ההודעות שלו באופן

מאובטח, באמצעות הצפנה כך שרק השרת והלקוח
אלו היתה אמורה להישלח ההודעה ידעו מה תוכן ההודעה ולא גורם שלישי עין.

מבנה שרת-לקוח:

שרת ראשי, שאליו ניתן להתחבר כמה לקוחות במקביל. השרת יודע לטפל בכל לקוח בנפרד, ולדעת לאיזה לקוח שייך איזה מידע. שימוש במודול select. Select לבד מבחין בין לקוחות שהתנתקו, ללקוחות שהתחברו, לקוחות מחכים להודעה ובעזרתו אפשר לשלוח כל הודעה
ללקוח הנכון.

אבטחת מידע של השרת:

השרת שומר מידע פנימי של הלקוח, כגון הסיסמא שלו, ההיסטוריה האישית שלו וגדומה אצלו במחשב בתיקייה אישית שאין אליה גישה מבחוץ, אלא רק השרת\המחשב ממנו מורץ השרת יכול לגשת לתיקייה. בתיקייה זו נשמרים שני דברים: הראשון, מידע רב שלא
ניתן\קשה לשמור אותו בבסיס נתונים מסוג sqllight3. זה כולל תיקייה לכל לקוח שם כל הקבצים שהעלה אותו לקוח נשמרים, מסודרים, נמחקים אם אין להם שימוש, ונעשה בהם שימוש של השרת. הדבר השני הוא database מסוג sqlight3 בו נשמרים דברים שלא נכון לשמור אותם בתיקיות ועליהם להיות שמורים גם במקרה שהשרת נופל לדוגמא. כגון שם משתמש וסיסמתו. כך בעצם כל המידע הנ"ל שמור ונגיש רק לשרת. לא ניתן באף דרך לפרוץ למחשב השרת וגנוב מידע.

מאגרי מידע:

כאמור, כל מאגרי המידע נמצאים על מחשב השרת. בתיקייה על מחשב השרת. תיקייה לכל לקוח בה יש תיקייה לכל אדם שאותו לקוח חיפש. בה יש 3 תיקיות: האדם המבוקש, גלריית תמונות, ותמונות בהם האדם נמצא. ובסיס נתונים בו יש את לכל לקוח את שם המשתמש
שלו, הסיסמא שלו, הpath של השרת בו נמצא המידע של הלקוח, האם הוא מחובר ותיקיות
היסטוריה שהשרת כבר העלה ללקוח.

אבטחת מידע העובר ברשת:

קיים מידע רגיש שאסור שגורם עין שלישי יקבל העובר ברשת. לשם כך קיים צורך שרק השרת והלקוח המסוים אליו מתכוון השרת לשלוח הודעה ידעו את המידע הנשלח. לשם כך היה צורך חיוני להצפין את המידע ברשת, בדרך שרק 2 הגורמים שציטו קודם יוכלו לפענח
אותו. החלטתי להשתמש בהצפנת RSA, אותה מימשתי בעצמי. בנוסף עוד הצפנות משנה.
כל ההצפנות מפורטות בהמשך.

תיאור הצפנות:

אלמוג גל. Face Founder.

RSA: כאשר השרת נהיה אקטיב, הוא קורא לפונקציה שיוצרת את המפתח הציבורי.
המפתח הציבורי נשמר כמשתנה גלובלי לאורך כל זמן הרצת השרת.

המפתח הציבורי בנוי משני מספרים: מספר ראשון- n- מכפלת שני מספרים ראשוניים רנדומליים. m- ביצוע פונקציית אולר (Euler) על המספר הראשון. מספר שני – לאחר יצירת m התחלה של חיפוש מספר החל מ5 (כל מספר ראשוני קטן עובד, בחרתי ב5) של מספר שגם לא זוגי וגם זר לחלוטין (אין להם מחלק משותף) מציאת המספר הנ"ל הינו המספר השני - e. הפתח הציבורי, כשמו הוא, נשלח לכל לקוח שמעוניין בו. לאחר מכן השרת יוצר מפתח פרטי שרק הוא יודע, זאת באמצעות אלגוריתם פשוט לשרת, אך לניחוש פורץ מאוד מסובך. אני בחרתי ליצור מספר k שיהיה שווה לאחד וגדיל את עצמו באחד בלולאה. בכל פעם בלולאה אני קורא לפונקציה בה אני מכפיל את k בm מוסיף להם אחד ומחלק בe. אם התוצאה שלמה הא המספר הפרטי של השרת – d.

כאשר לקוח מעוניין להתחבר הוא מקבל את המפתח הציבורי. לאחר מכן רק לקוח שיודע את הפרוטוקול של השרת (אם כי עדיין בעל סיכוי לניחוש\פריצה) יודע שהוא צריך ליצור בעצמו עוד מספר. יצירת המספר של הלקוח היא יצירת מספר רנדומאלי (באמצעות Random) בין המספר הציבורי השני שקיבל לבין מספר היכולת הרצה של המחשב שלך (כמה שהכוח המעבד שלך קטן יותר כך המספר יקטן). מכיוון שאני עובד במחשב ביתי, ואין ברשותי כוח מעבד כמו חברות אבטחה גדולות שמגיעות גם למספרים של 50 ו60 ספרות, מספר
ההגבלה שלי הוא 150000. הגעתי למספר זה באמצעות לולאה ששיניתי כל פעם את המספר עד שהגעתי למספר ההגבלה הגדול ביותר וזמן ההרצה הקטן ביותר (לא יותר מכמה שניות במקרה הגרוע ביותר). לאחר יצירת המספר הנ"ל הלקוח מעלה את המספר שיצר בחזקת המספר הראשון הציבורי והשארית של התוצאה מהמספר השני הציבורי תיתן את המפתח הסימטרי הפרטי. הלקוח שולח את המספר הרנדומלי (לא הפרטי) שמצא לשרת.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה הא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הוא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות
ביניהן כך שרק הם יודעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה (שאפרט בהמשך) ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין
ברשת, בלי המספר הפרטי של השרת d הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של
הלקוח והשרת.

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח
והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים יודע מה המידע בהודעה. אלגוריתם ההצפנה בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה (הודעה) מסוג
רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו (באמצעות פונקציית ord). את המספר הנ"ל הופכים (נגד 1546 הופך
ל6451). מחסירים ממנו 100 עד שנמצא בין 0 ל255, בנתיים שנו מונה (ערכו הראשוני 40) שעולה ,בכל החסרה של המספר ההפוך, במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה
למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג
התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח

אלמוג גל. Face Founder.

יהיה מסוגל להעביר את האותיות הascii למקור, שכן הוא יודע כמה)בעזרת ציוף המונה ליד כל אות(פעמים להכפיל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק מידע מוצפן. גם בעזרת brute force לא ימצא מה הפאקטה המקורית כי אין ברשותו את המפתח. כאמור הוא יכל לנחש פעם את המפתח ועלול לנסות brute force, אבל כמובן דבר זה ייקח לו שנים. חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי מאלפי תווים אבל זוהי בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהיא רשימה, עוברת pickle, שהופך אותה לstring ארוך. דבר שמקשה עוד יותר על גורם שלישי עין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם רק השרת והלקוח יודעים את ההודעות שהם שולחים זה לזה.

תיאור אלגוריתם ראשי:

שרת: שרת מורץ בלולאה תמידית ומחכה לחיבור של לקוחות, אותם הוא שומר. השרת רק מקבל הודעות ופועל בהתאם. השרת מנוט את המידע שלו בין הלקוחות ושולח להם אותו, או מקבל אותו בהתאם ועושה פעולות הקשורות ללקוח בהתאם. בנוסף השרת תומך בלקוח הכי "טיפש" שיכולתי לבנות, ז"א שרוב עניין האלגוריתם בקשר לתמונות, מכונות וכו' נמצא אצל השרת.

לקוח — תפקיד האלגוריתם אצל הלקוח הוא לדאוג לזרימה מסויימת של הלקוח באפליקציה על פי סדר כרונולוגי. למשל, לנוט את השרת בין חלון לחלון הבא המתאים לו, לדעת שבתחילת הקוד יש לדאוג שהלקוח יבקש מפתח ציבורי וכדומה. השרת מנוט כך את הלקוח שרואה זאת ויואלת באמצעות GUI שגם האלגוריתם שלו שייך לשרת. בעזרת הGUI הלקוח גורם למשתמש לנוע ברחבי באפליקציה, באופן מסודר ושניתן לפקח עליו ולהראות לו מידע וחלונות בהתאם. השתמשתי בGUI של בtkinter. בנוסף, באמצעות הGUI הלקוח מקבל קלטים מהמשתמש, שומר ומעבד אותם אצלו ושולח בהתאם הודעה לשרת, ואז מצג את תשובת השרת. הלקוח דואג גם להציג תמונות ללקוח, וכך בעצם גם לשמירת מידע, תמונות, על מחשב הלקוח באופן הכי יעיל ושלא ישימו אליו לב.

הלקוח הוא בעצם זה שדואג לעניינים החזותיים והנוחות בעוד שהשרת על האלגוריתמיקה הטכנית.

הקשרים בין יחידות שונות: ניתן לראות תחת Top-Down-Level-Design של ארכיטקטורת הפרוייקט.

הצגת מקרים)use cases(עבור פונקציות עיקריות:

ניקח שתי פונקציות עיקריות בקוד שלי: 1. הכנסת האנשים המתאימים המקבלת את שם המשתמש, שם התמונה, ומידע בינארי של התמונה.

```
def insert_correct_images(username, name, data):
```

קיים משתנה גלובלי בתוכנית שלי אותו אני מאפס אחרי כל תהליך והוא מערך של התמונות הנכונות)correct_images(. הפונקציה הנ"ל יוצרת path במקום שנועד לאפליקציות בwindows בשם של השעה והתאריך המדויק. לאחר מכן באותו path יוצאת קובץ התואם לשם הקובץ ושומרת אותו. הפונקציה כותבת בינארית בו את כל המידע שקיבלה. הפונקציה סוגרת את הקובץ ושומרת את התמונה במערך התמונות הנכונות. לסיום שולחת לשרת

שקבלה את התמונה שבה האדם נמצא, ושהיא מחכה לעוד תמונות. ראה דוגמא תחת use case diagram, עמוד 20.

2. פונקציית "חלון הקלט". פונקציה שמקבלת את שם המשתמש, חלון הרקע - שערכו הדיפולטיבי הוא תמונה של חלון הקלט, משתנה בוליאני אדם - שערכו הדיפולטיבי שקר
ומשתנה בוליאני גלריה — שערכו הדיפולטיבי שקר.

```
def input_window(username, background=Inputs_B,
person=False, gallery=False):
```

הפונקציה מוחקת את כל הכפתורים והרקעים שהיה בהם שימוש קודם לכן, כדי לא לבזבז
מקום מיותר ועבודת מעבד מיותרת. הפונקציה מציגה את הרקע של התמונה שקיבלה, הרקע הראשוני של חלון הקלט או רקע במקרה שהא כבר בחר אדם או כבר בחר אדם
וגלריה. הכותרת של החלון תהיה Input Window.

כאשר משתנה האדם הוא שקר סימן שהמשתמש לא הכניס אדם. לכן יופיע לו כפתור של "בחר אדם". אליו יכנס ויבחר אדם. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור
לתחילת תהליך חלון הקלט.

כאשר המשתנה אדם נכון, סימן שהמשתמש כבר בחר אדם ולכן הרקע שיוצג לו הוא שכבר בחר אדם. בנוסף אם אדם סימן שלא נבחרה עוד גלריה ולכן יופיע לו כפתור "בחר גלריה". אליו יכנס ויבחר גלריה. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור לתחילת
תהליך חלון הקלט, שגם אדם הוא שקר.

לאחר מכן אם גלריה נכון, וגם אדם נכון יוצג רקע שהמשתמש סיים את תהליך הקלט. יופיע לו כפתור "המשך לתהליך" שלחיצה עליו תמשיך בתהליך מצאת הפנים. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור לתחילת תהליך חלון הקלט שגם אדם שקר וגם גלריה
שקר.

בנוסף לאורך כל המצבים ישנה דאגה לevent בו המשתמש סוגר את החלון, מצב יציאה,
שיקרה לפונקציית יציאה שתדאג ליציאה מסודרת.

עץ מודולים:

Server_Functions.py:

```
import select
import pickle
from Server.Preparation_For_Encyption import *
from Server.Encryption_And_Decryption import *
import base64
import os.path
import os
import matplotlib.pyplot as plt
import datetime
import ntpath
# connection setting:
from Server.Communication_Settings import *
import socket
import glob
from Server.Detect_faces import *
import shutil
```

Preperation_For_Encryption.py:

```
from random import randrange, getrandbits
import time
```

Detect_Faces.py:

```
from_future_import print_function
import click
import os
import re
import face_recognition.api as face_recognition
import multiprocessing
import itertools
import PIL.Image
import numpy as np
import ntpath
```

Database.py:

```
import sqlite3
import os
import psutil
```

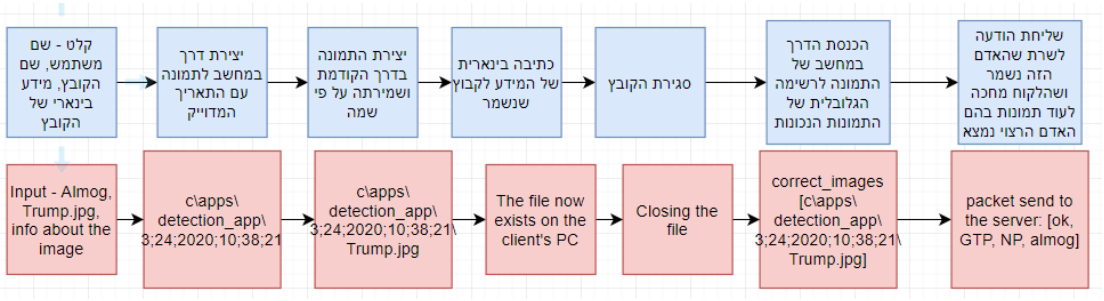
Handle_Images.py:

```
import sqlite3
import os
import psutil
```

Client_Functions.py:

```
from tkinter import DoubleVar, filedialog, ttk
#StringVar
import socket
import pickle
from Client.Encryption_And_Decryption import *
import random
import base64
import ntpath
import matplotlib.pyplot as plt
import datetime
from PIL import Image, ImageTk
import time
# connection setting:
from Client.Communication_Settings import *
from Client.Face_Detection_And_Comparison import *
from Client.Handle_Images import *
import shutil
```

Use case diagram:



רשימת use cases:

ישנם שלושה חלקים עיקריים לפרויקט: פתיחה- הרשמה וכניסה למערכת, הכנסת קלט של המשתמש, הצגת פלט למשתמש. ניתן use case לכל אחד מהחלקים האלו:

1. משתמש לוחץ על כניסה למערכת ומופיע לפניו שתי שורות לכתיבה. אחת שם המשתמש ואחת הסיסמא. נניח שהמשתמש טועה בכתיבת הסיסמא שלו ולוחץ על כפתור "הבא"\ לוחץ על כפתור המקלדת enter. המידע יועבר מן הGUI אל הלקוח. הלקוח ישלח את שם המשתמש והסיסמא בפאקטה לשרת. ההודעה תוצפן והשרת יפענח אותה. השרת יבדוק

ברשימת הsql שלו האם ישנו אותו שם משתמש שהכנס תחת העמודה "שם המשתמש". כשאר ימצא, דבר שיקרה כי המשתמש טעה בסיסמתו, השרת יעבור לבדוק מהי הסיסמא המתאימה לשם המשתמש שמצא, ואותה ישווה לסיסמא שקיבל מהלקוח. כאשר ישווה ויראה שהסיסמא שנשלחה ע"י הלקוח לא תואמת לסיסמא בטבלת הsql תחת אותו משתמש ויצא השרת הודעת error. בהודעה ירשום השרת ללקוח ששם המשתמש או הסיסמא שגויים. ההודעה תוצפן ותפוענח ע"י הלקוח. הלקוח יראה שההודעה מוסרת ששינה טעות ואחד מהנתונים שהכניס הלקוח שגויים. לכן הלקוח יחליף את הרקע של מסך הכניסה למשתמש, במסך)שעוצב מראש על ידי(שבו רשום בהודעה אדומה ששם המשתמש או הסיסמא שגויים. השרת ישמור את שם המשתמש וסיסמא השגויים וינקה אותם לשורות

בהתאם, למקרה למשל כמו זה, בו ישנה טעות קטנה בהקלדת הסיסמא. כך הלקוח ידע על הטעות וגם לא יצטרך לרשום את פרטיו מחדש אלא פשוט לתקן את הטעות בסיסמתו, להזין

אותה שנית, וכל התהליך חוזר, עד אישור השרת לכניסת המשתמש למערכת.

2. משתמש מגיע למסך הקלט. תחילה עליו להכניס תמונה של אדם אחד)לא יותר ולא פחות(כקלט שאותו ירצה למצוא לאחר מכן בתוך גלריית תמונות. המשתמש בטעות לחץ על

התמונה הלא נכונה, ולחץ על התמונה שבה הרבה אנשים. התמונה תחילה, אצל הלקוח, ממורת למידע בינארי. המידע הנ"ל נשלח לשרת תחת בקשה של יצירת תיקייה חדשה

לאותו אדם, אליה יועלה מידע בהמשך. המידע מוצפן ומפוענח ע"י השרת. השרת ממיר את המידע הבינארי ושומר את התמונה שנשלחה אצלו בשרת.

השרת מעביר את הpath של התמונה למכונה הלומדת שבודקת האם יש פרצוף אחד בתמונה. המכונה תחזיר שבתמונה

יש כמה פרצופים. השרת ישלח הודעת error ללקוח שבתמונה יש יותר מפרצוף אחד. ההודעה תוצפן ותפוענח ע"י הלקוח. הלקוח יקבל את ההודעה וישנה את מסך הרקע בחלון הקלט של האדם אותו תרצה לחפש, לרקע שבו צצה הודעת שגיאה לפיה בתמונה יש יותר מפרצוף אחד, ועל המשתמש לחפש אדם אחד. המשתמש יראה את ההודעה)כמו כן, תנאי

זה הופיע לו בעמוד ההוראות שעבר בו קודם לכן(, יבין שלחץ על התמונה הלא נכונה. תהיינה לו האפשרות לבחור תמונה שוב, והפעם יבחר המשתמש בזהירות את התמונה

אליה התכוון מלכתחילה. התהליך חוזר, והפעם השרת יאשר את התמונה שהועלתה והלקוח יוכל להמשיך במסך הקלט שלו)יצטרך להכניס גלריה(.

אלמוג גל. Face Founder.

3. המשתמש סיים את התהליך. כרגע מוצגות בפניו התמונות בהם האדם שחיפש נמצא ושתי אפשרויות, בנוסף המידע של התהליך הזה נשמר כבר אצל השרת. כפתור "לעשות את התהליך מחדש" אשר לחיצה עליו תגיע ללקוח והוא יחזיר את הלקוח למסך הקלט. וכפתור "עבור להיסטוריה". לחיצה על כפתור זה תגרום ללקוח לשלוח בקשת העלאת קבצי

היסטוריה לשרת. ההודעה תוצפן ותפוענח ע"י השרת. השרת יקבל את הפאקטה, בה רשומים פרטי המשתמש. באמצעותם ילך לשורת המשתמש בטבלת הsql ושם לעמודת

תיקיות היסטוריה. השרת ינסה לרשימה את כל התיקיות הנ"ל. השרת יקבל את הpath של הלקוח גם בעזרת פרטיו וטבל הsql שברשותו. כך ידע לאן לגשת ולהוציא את כל תיקיות

ההיסטוריה שקיימות לאותו משתמש במחשב לו. תיקיות אלו מעולות ישר לשרת, אך לא ישר ללקוח. בעזרת חיסור הרשימות ידע השרת אילו תיקיות עוד לא העלה ללקוח ואת שמם ישלח ללקוח עם הוראה ליצור תיקיות אלו בpath ההיסטוריה של הלקוח. ההודעה תוצפן

ותפוענח ע"י הלקוח. הלקוח באמצעות משתנה גלובלי יודע איפה ההיסטוריה של המשתמש שמורה אצלו במחשב. שם, לפי השמות שקבל, ייצור תיקיות היסטוריה בהתאם. לאחר סיום ייצור ההיסטוריה ישלח הלקוח שסיים לעלות את תיקיות ההיסטוריה. ההודעה תוצפן

ותפוענח ע"י השרת. השרת יקבל את ההודעה ויתחל לשלוח קבצי היסטוריה רלוונטיים. השרת דאג לזכור את התיקיות החדשות, וקדם שומר שהעלה אותם בטבלת הsql תחת אותו שם משתמש תחת עמודת תיקיות היסטוריה. לאחר מכן עבור על קובץ בתיקיות הנ"ל

שולח השרת את המידע של הקובץ, ואיפה לשמור אותו ללקוח. לאחר מכן מחכה שנייה אחת כדי שהלקוח יעמוד בקצב. הלקוח כל הודעה מקבל ומפענח וכך הוא יודע איזה תמונה ואיפה לשמור בהיסטוריה. לאחר העלאת כל התיקיות מוצגת ספריית ההיסטוריה למשתמש, שם הוא יוכל לעיין, למחוק, להעתיק וכו' קבצים.

תרשים UML:

Class: DATABASE

Operations	Values	מה הפעולה עושה
Databse	db_location	פותחת קובץ מסוג db. הקובץ הינו מבנה נתונים, טבלה מסוג sql, שניתן לשמור בה מידע
fetch		מחזירה את כל הנתונים בטבלה.
insert	username, password	מחזירה אמת אם לא קיים כזה שם משתמש ושם המשתמש והסיסמא הוכנסו בהצלחה, אחרת שקר.
exists	username, password	מחזירה אמת אם קיים שם משתמש שתואם לסיסמא, אחרת שקר
generate_path	username	יוצרת path להיסטוריה של המשתמש ושומרת אותו, ומחזירה אותו
active_user	username	אם משתמש לא אקטיבי, הופכת אותו לאקטיבי ומחזירה אמת, אחרת שקר

unactive_user	**username**	**הופכת משתמש ללא אקטיבי, אם הפעולה בוצעה בהצלחה מחזירה אמת**
remove_user	**username**	**מוחקת מהטבלה שם שורה שבה נמצא שם המשתמש**
unactive_all		**הופכת את כל המשתמשים בטבלה ללא אקטיביים**
update_user	**username, password**	**מעדכנת את הסיסמא של שם המשתמש, במידה וקיים**
running		**מחזירה true אם הטבלה מורצת מcmd או מpython**
get_user_path	**username**	**מחזירה את path ההיסטוריה של שם המשתמש**
insert_user_history_folder	**username, folder_name**	**מוסיפה ושומרת שם נוסף של תיקיית ההיסטוריה בעמודת "תיקיות ההיסטוריה"**
get_history_folders	**username**	**מחזיר את כל תיקיות ההיסטוריה ששמורות תחת שם המשתמש**
del		**מוחקת את כל הטבלה**

רכיבי ממשק:

ממשק המשתמש הגרפי, הGUI, נכתב כולו בtkinter. Tkinter הינו ממשק משתמש גרפי בשפת הpython באמצעותו ניתן לעצב חלונות, כפתורים, רקעים, תאי מילוי, פס גלילה, מד טעינה, הצגת תמונות ועוד. מצאתי זאת לנוח לכתוב את כל הפרוייקט בשפה אחת – python ולכן נבחר. רוב הלקוח שלי הוא קוד tkinter. בעיצוב האפליקציה השקע הרבה שורות קוד, חשיבה וזמן.

מדריך למשתמש:

על המשתמש להוריד ספריית קבצים בשם "Client" לאותו הpath במחשב שלו. שם יהו לו 6 קבצים\תיקיות:

תיקיה ראשונה- Design – שם יהו כל עיצובי הרקע והGUI של האפליקציה. קובץ Client_Functions.py – שם יהו כל פונקציות הפייתון הקשורות לתפעול הלקוח, קובץ הלקוח המרכזי שבסופו של דבר מורץ.

קובץ Communication_Setting.py – קובץ שם יהו ההגדרות הקשוחות לחיבור הלקוח והשרת, הגדרות שצריכות להיות זהות לזה של השרת.

קובץ Encryption_And_Decryption.py – קובץ בו 2 פונקציות הקשורות לאבטחת המידע של הלקוח: הצפנה ופיענוח. גם קובץ זה צריך להיות זהה בצד השרת, כדי שיצפנו ופענחו באופן זהה.

קובץ Handle_Images.py – קובץ האחראי על טיפול הצגת התמונות במסכי הGUI השונים באפליקציה.

קובץ Detection_app.exe קובץ אשר ירץ, ע"י הפעלתו בלחיצה, את הקובץ Client_Functions.py אשר יפעיל את שאר הקבצים לפי הקוד הרשום בו.

לאחר הורדת כל הקבצים הנ"ל יש ללחוץ על הקובץ Destection_app.exe ובכך יורץ הלקוח ויחד אתו האפליקציה.

* לאורך כל התהליך באפליקציה בהמשך יש למשתמש אפשרות נוספת שהיא לסגור את חלון האפליקציה, סגירת החלון תובל ליציאה לא מסודרת מהאפליקציה ותסגור את החלון והתוכנה. ביציאה זו המשתמש עלול לאבד מידע.

בפני המשתמש יופיע מסך הפתיחה הבא:

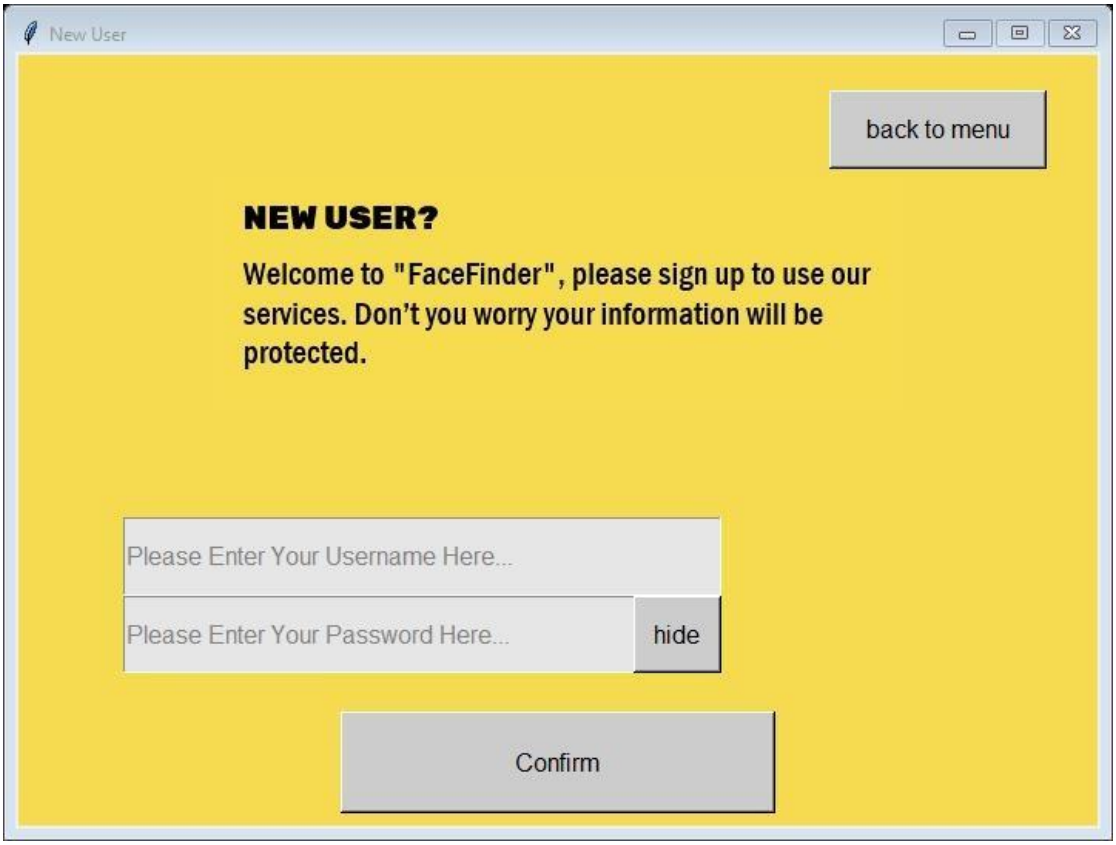
אלמוג גל. Face Founder.

ניתן לראות בתמונה את מסך הפתיחה, עם זכויות יוצרים על יד, אלמוג גל, והלוגו המעוצב של האפלקציה. בפני המשתמש שתי אפשרויות:

כפתור הlogin יוביל את המשתמש למסך בו יצטרך להכניס פרטי כניסה למשתמש קיים.

כפתור הnew user יוביל את המשתמש למסך בו יצטרך להכניס פרטי כניסה, שאותם יצטרך לזכור, ולפתוח משתמש חדש באפלקציה.

נניח שהמשתמש פעם ראשונה משתמש באפלקציה, ועל כן ילחץ על כפתור הnew user.



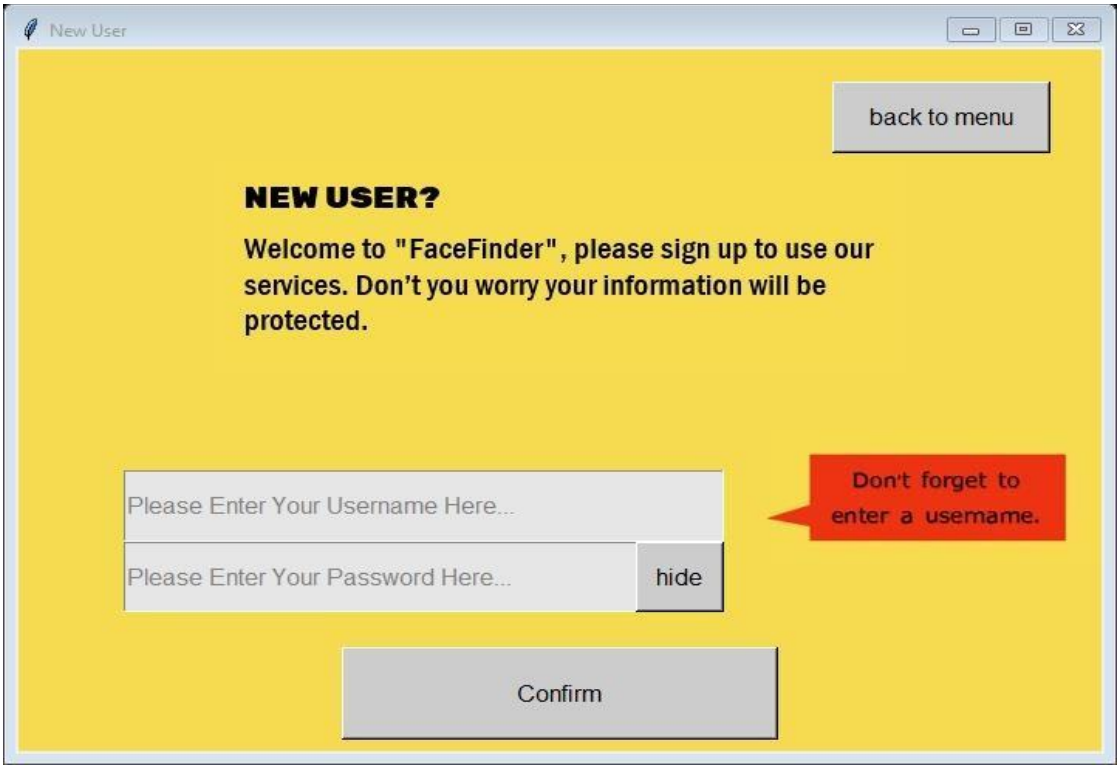
ניתן לראות את מסך יצירת משתמש חדש באפלקציה. בפני המשתמש מספר אפשרות.

לחיצה על כפתור back to menu תחזיר את המשתמש למסך הפתחה)ראה עמוד 24(.

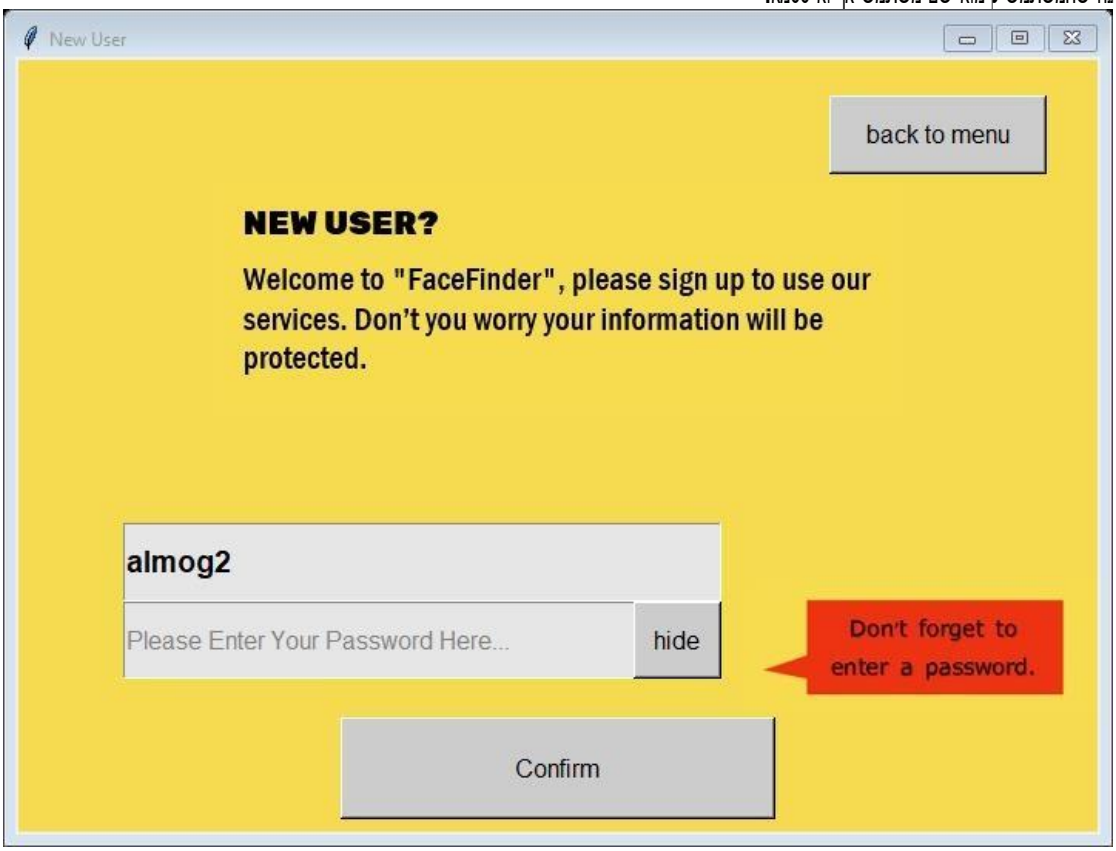
על המשתמש להכניס פרטים תקיים, שהוא את מפתח הכניסה שלו למשתמש אישי. נניח שהמשתמש לא מלא אף תא. מכון שהמשתמש לא

מלא שם תא שם המשתמש האפלקציה
דואגת להזכיר לו זאת.

אלמוג גל. Face Founder.



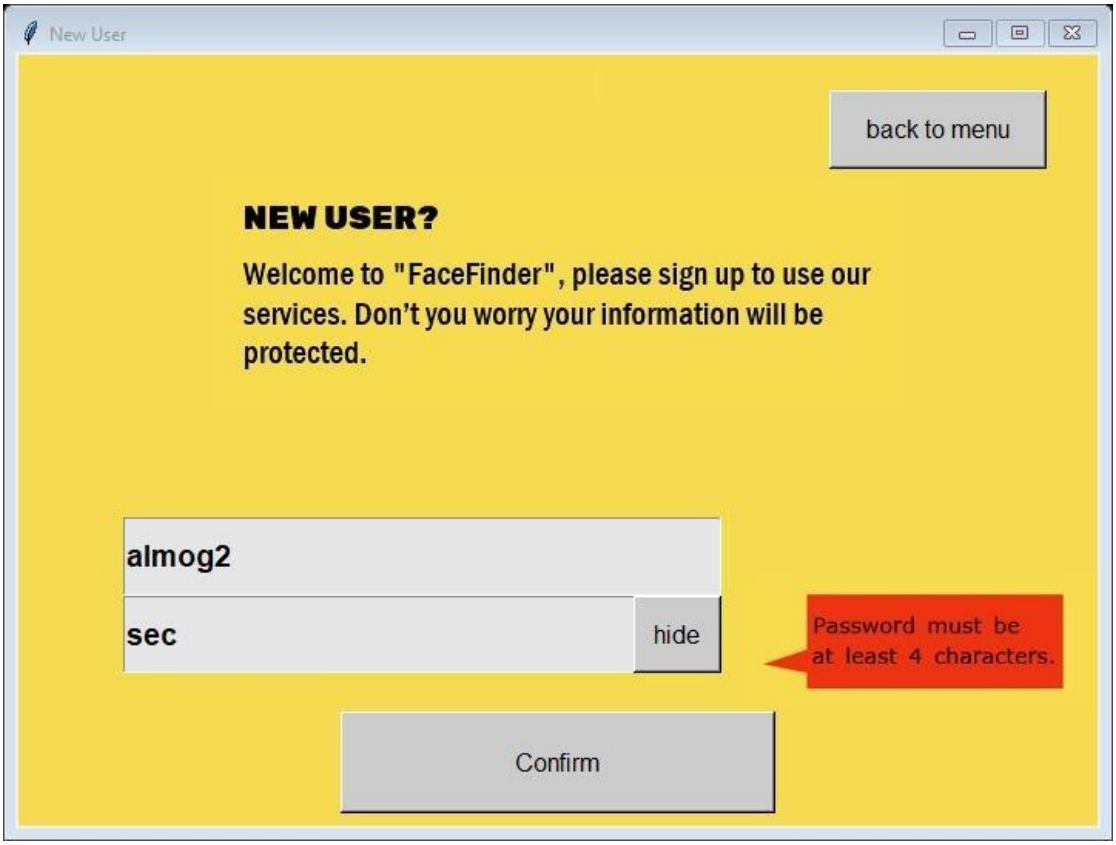
מצב שהמשתמש כן מלא שם משתמש אך לא סיסמא.



מצב שהמשתמש מלא שם משתמש קצר מיד.

אלמוג גל. Face Founder.

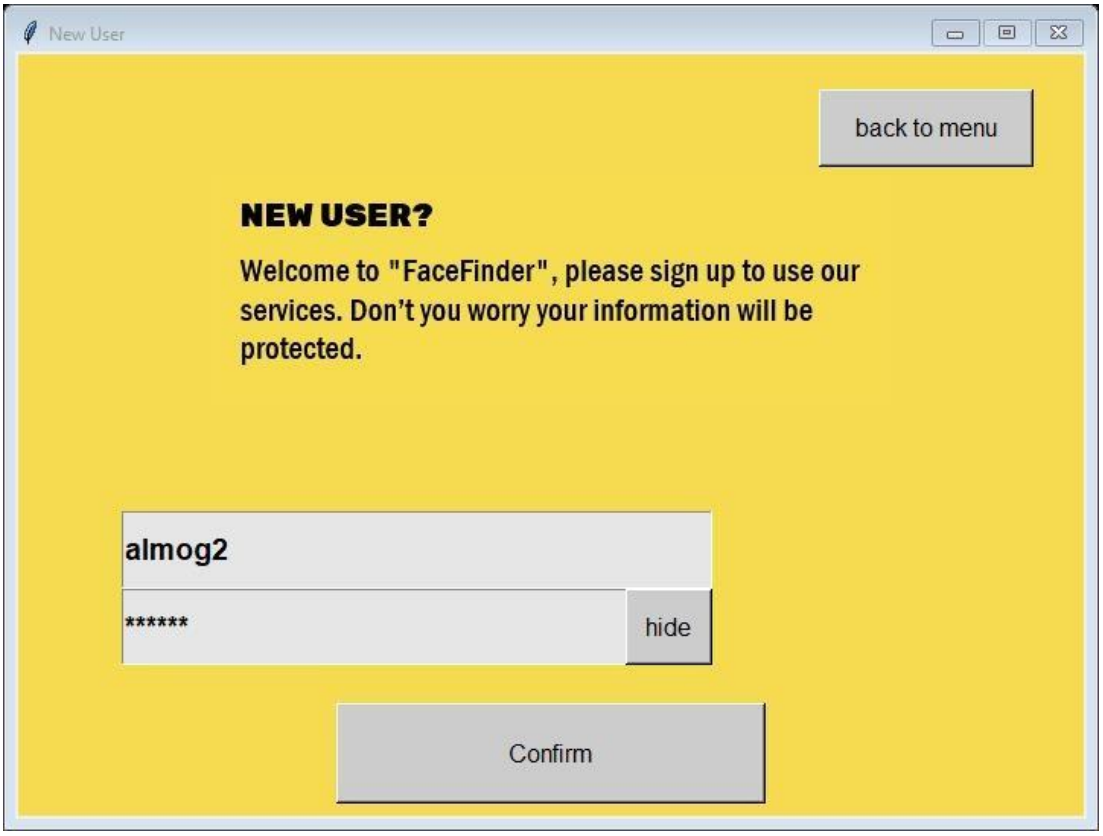
נניח שהמשתמש מילא סיסמא קצרה מיד.



נניח שהמשתמש מילא שם משתמש שכבר קיים במערכת, ונמצא בשימוש.

אלמוג גל. Face Founder.

נניח שהמשתמש נמצא במקום ציבורי ולא מעוניין לראות את סיסמתו, לכן לוחץ על כפתור הhide.



אלמוג גל. Face Founder.

לאחר הכנסת שם משתמש וסיסמא תקינים ולחיצה על כפתור הconfirm יחזור המשתמש למסך הפתיחה ויקבל הודעה שהרשמתו בוצעה בהצלחה. ועלי להכנס לחשבון שלו עם הפרטים שלו כדי
להשתמש באפלקצה.

שוב, חזרה למסך הפתיחה)ראה עמוד 24(.

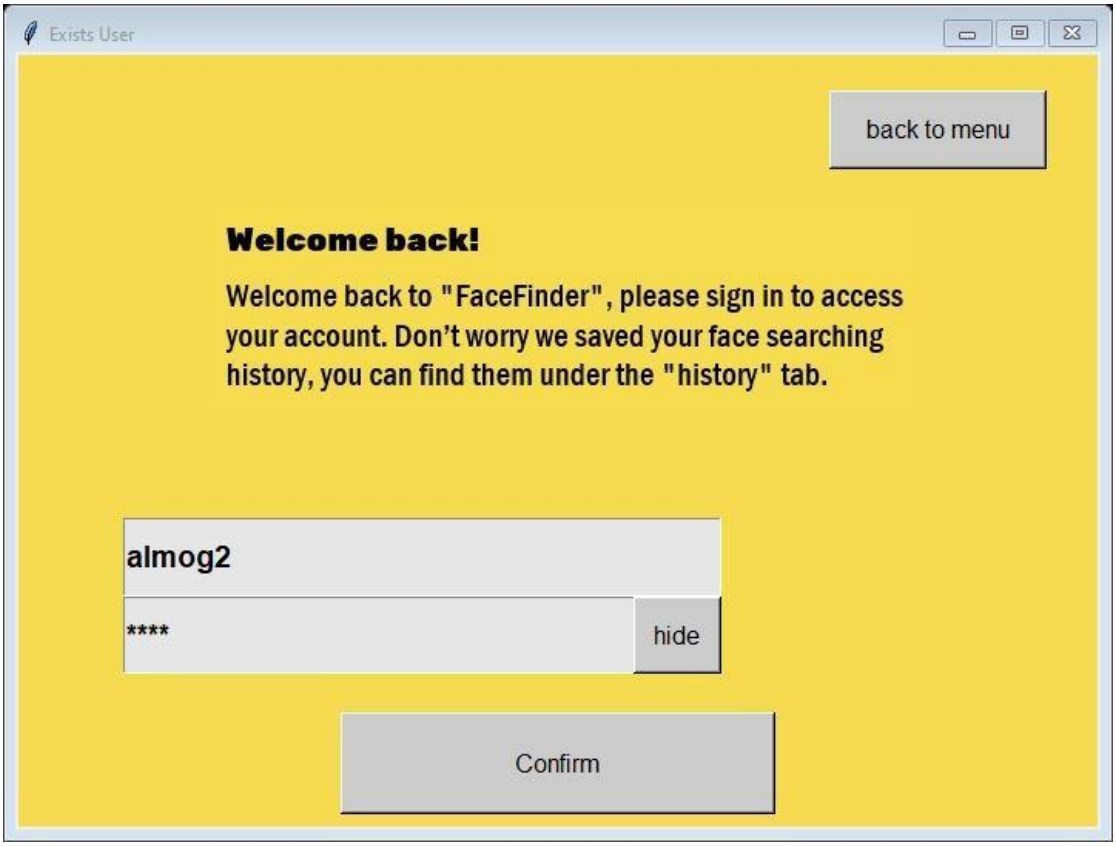
כעת המשתמש רוצה להכנס לחשבון שלו. לכן הוא לוחץ על כפתור הlogin.

בפני המשתמש אפשרות.

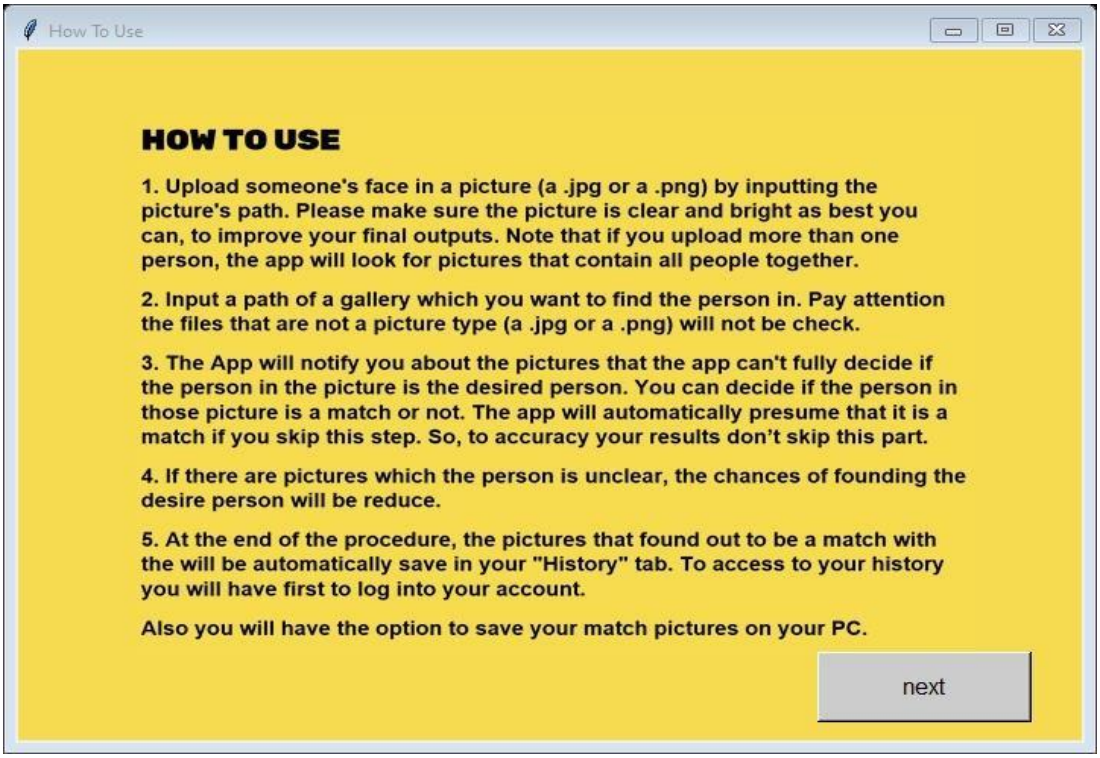
לחיצה על כפתור הback to menu תחזיר אותו למסך הפתיחה)ראה עמוד 24(. מיוח שהמשתמש הכניס פרטים שגויים –

שם המשתמש לא קיים במערכת או הסיסמא לא קשרה
לשם המשתמש.

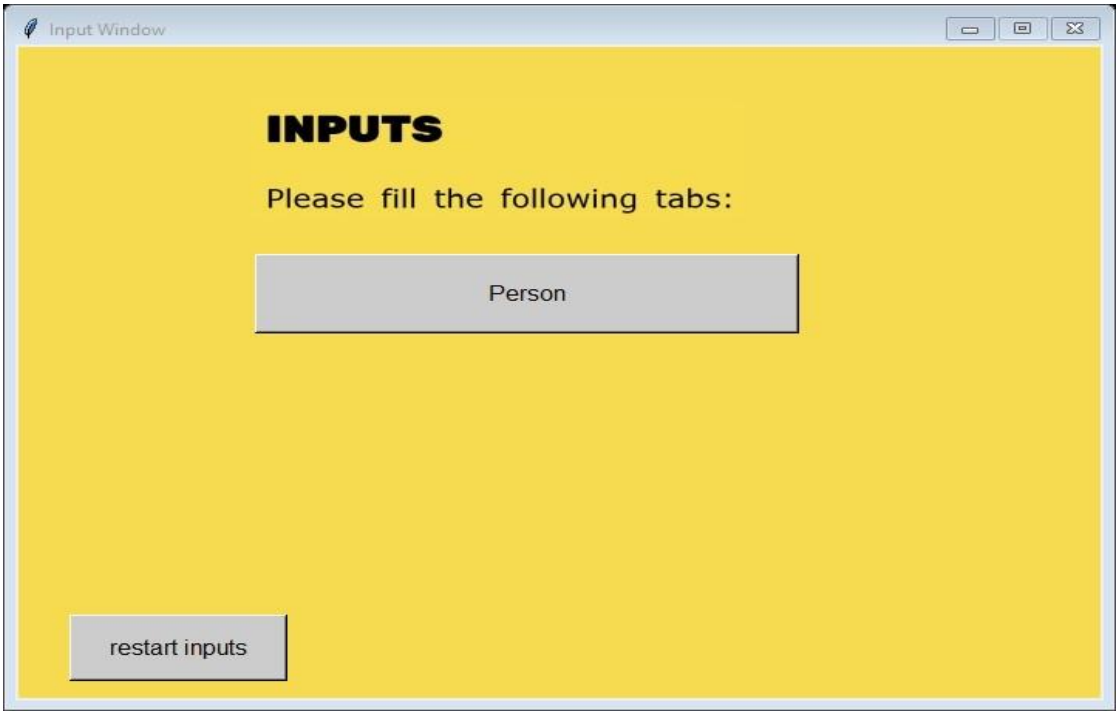
נניח שהמשתמש נמצא במקום ציבורי ולא מעוניין שיראו את סיסמתו.



המשתמש מכניס כעת פרטים תקינים ומכניס ולחץ על confirm ועבר למסך הבא. המסך הבא הינו מסך הוראות שנועד להסבר למשתמש כיצד להשתמש באפליקציה.)הערת כותב: החלון לא עדכני, ההוראות שונו ומסך שונצב מחדש בהתאם ברגע שההוראות יהו סופיות(.



לאחר עיון רב בהוראות המשתמש הפנים כיצד להשתמש באפלקציה, ומה השמוש של האפלקציה
ולחץ על כפתור הnext כדי להמשיך הלאה.

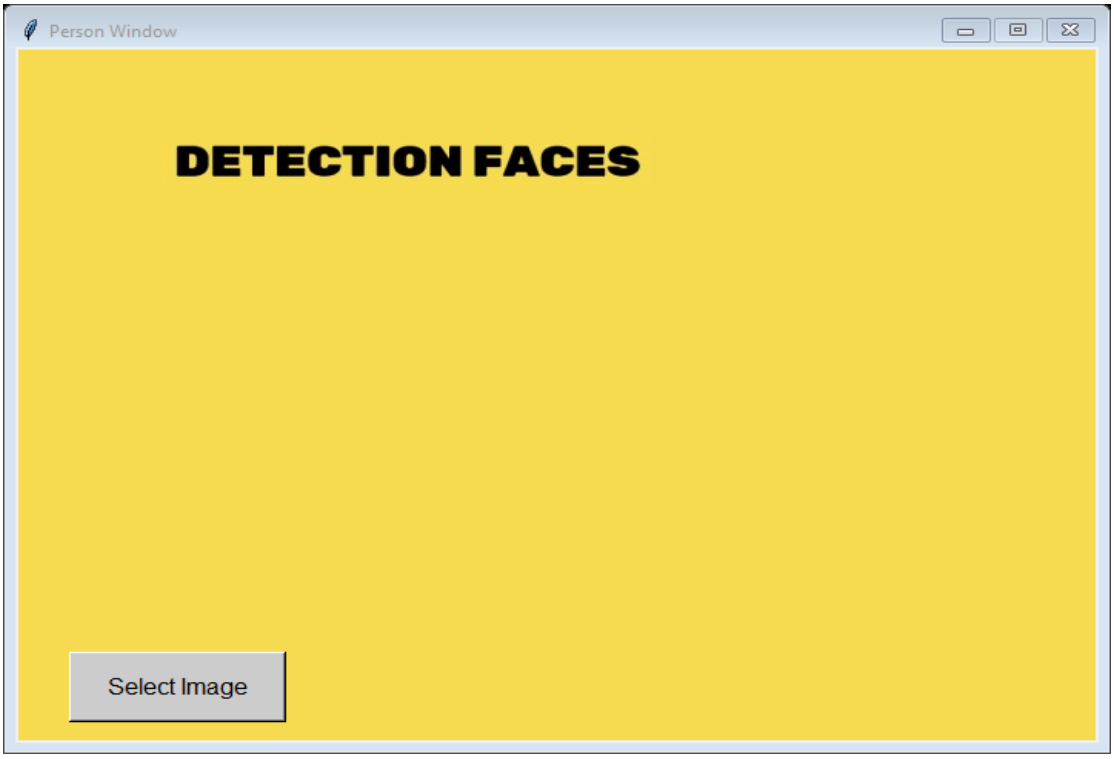


בפני המשתמש מסך קלט. לפני המשתמש כמה אפשרות:

לחיצה על כפתור back שתחזיר את המשתמש להוראות האפלקציה)כפתור זה עדיין לא שוצב +
ראה עמוד 31(.

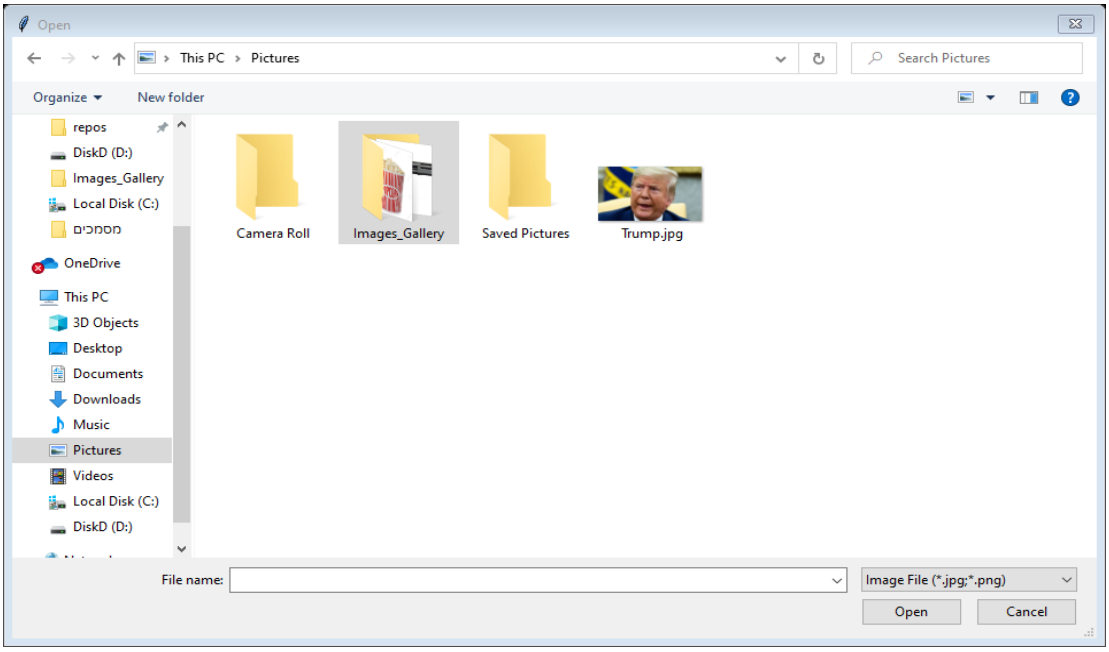
לחיצה כל כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש,
ותחזיר אותו למסך הקלט)ראה עמוד 32(.

על מנת להשתמש ולחפש אדם מסוים מתוך גלרית תמונות המשתמש לוחץ על כפתור הPerson.



אלמוג גל. Face Founder.

במסך "זיהוי פנים" זה המשתמש לחץ על כפתור select image כדי לבחור תמונה של האדם אותו רוצה לחפש.



לחיצה על הכפתור פתחה דיאלוג של המשתמש עם File Explorer דיאלוג של הלקוח עם ה File Explorer. על הלקוח לבחור תמונה בה מופיע האדם שהוא מחפש. בנוסף, הדיאלוג מגבל את המשתמש לבחור רק קבצי jpg. או png. . ועל כן, הוא לא יכל לבחור קובץ מסוג לא נכון.

לחיצה על כפתור האיקס וסגרת הדיאלוג תובל את הלקוח למסך "זיהוי פנים" (ראה עמוד 32). לחיצה על תמונה בה יש יותר מפנים אחד תראה למשתמש חלון עם הערה שהוא הכניס תמונה עם יותר מפנים אחד (התמונה עדיין לא עובדה).

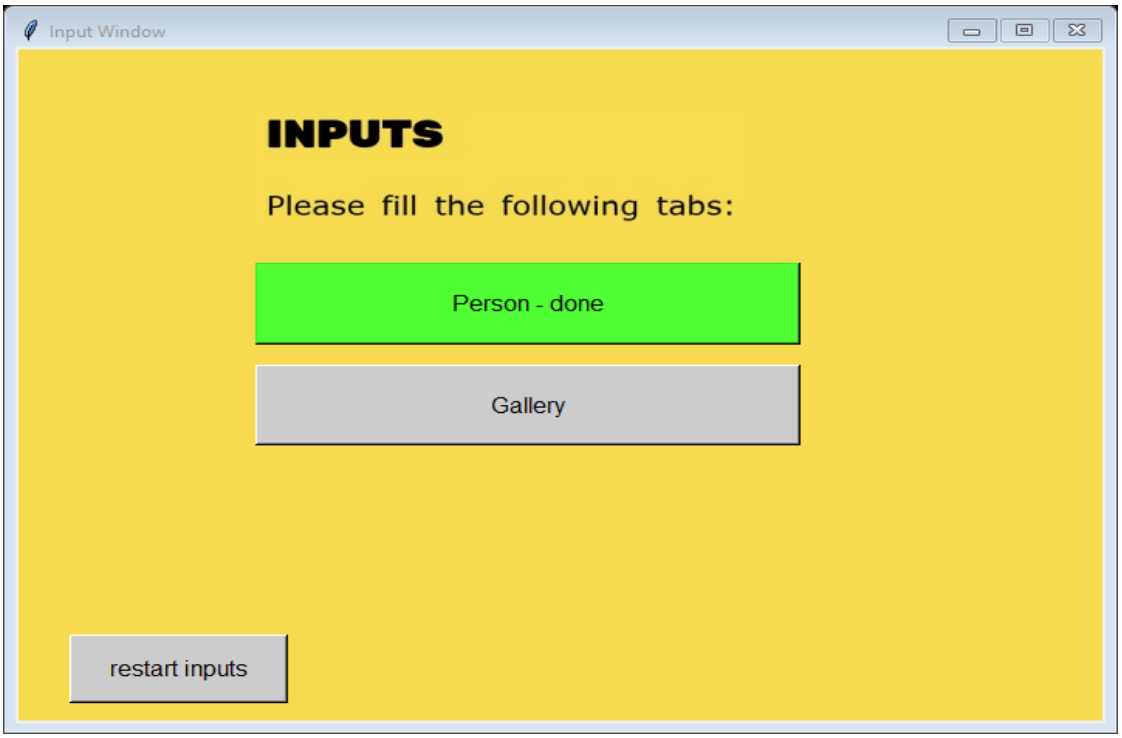
לחיצה על תמונה בה יש פחות מפנים אחד תראה למשתמש חלון עם הערה שהוא הכניס תמונה עם פחות מפנים אחד (התמונה עדיין לא עובדה).

המשתמש בוחר תמונה אותה הוא חצה לחפש (נניח, Trump.jpg).

ללקוח יפתח החלון ב יכל לראות בברור איזה בן אדם הוא בחר. בפני המשתמש שתי אפשרויות:

ללחוץ על כפתור change image שתובל אותו לדיאלוג חוזר על הFile Explorer)ראה עמוד 33(.

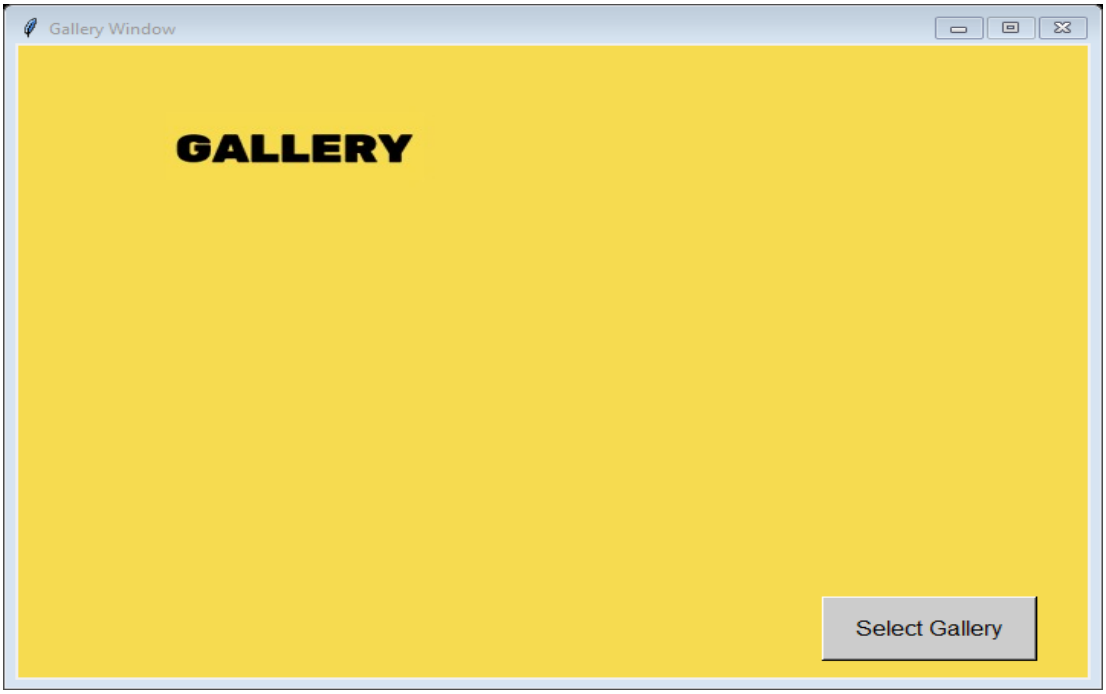
ללחוץ כפתור next מה שאומר שזה האדם שברצונו לחפש.



לאחר שהמשתמש בחר אדם, האפלקצה מודיעה לו שאת החלק הזה הוא סיים ועלו עכשו לבחור גלרה.

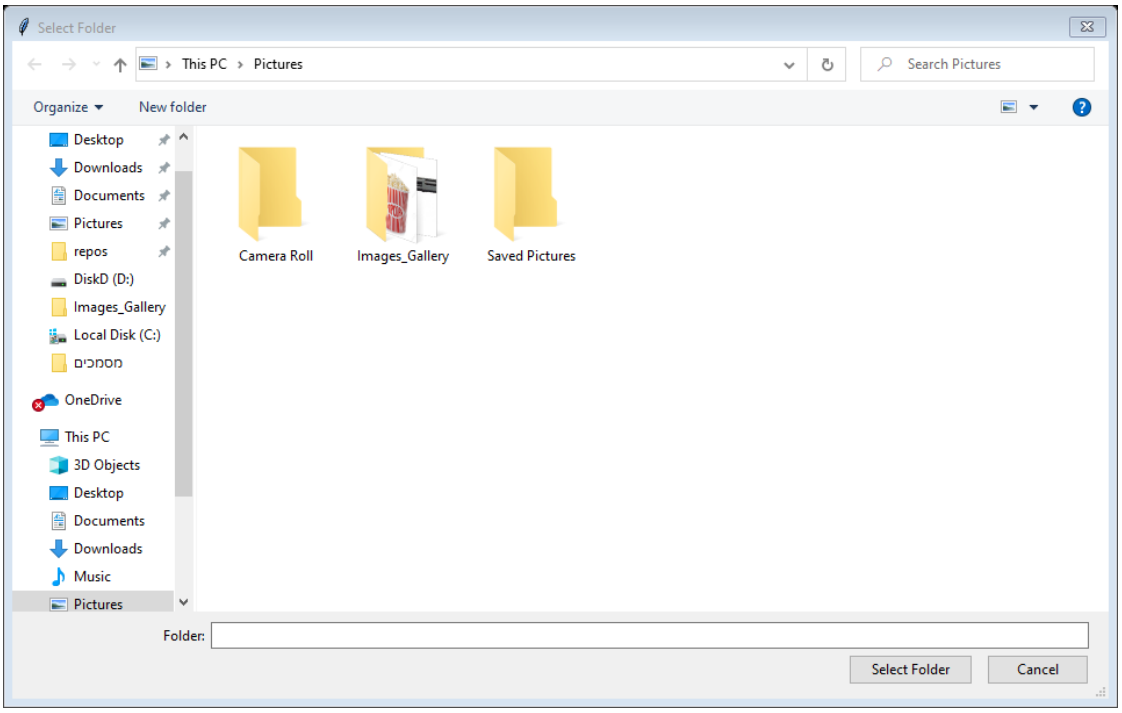
לחיצה כל כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש, ותחזיר אותו למסך הקלט)ראה עמוד 32(.

המשתמש רוצה להמשיך בתהליך ולכן לוחץ על כפתור gallery.



אלמוג גל. Face Founder.

נפתח חלון "גלרה", בפני המשתמש כפתור select gallery כדי להתחיל לבחור גלרה.

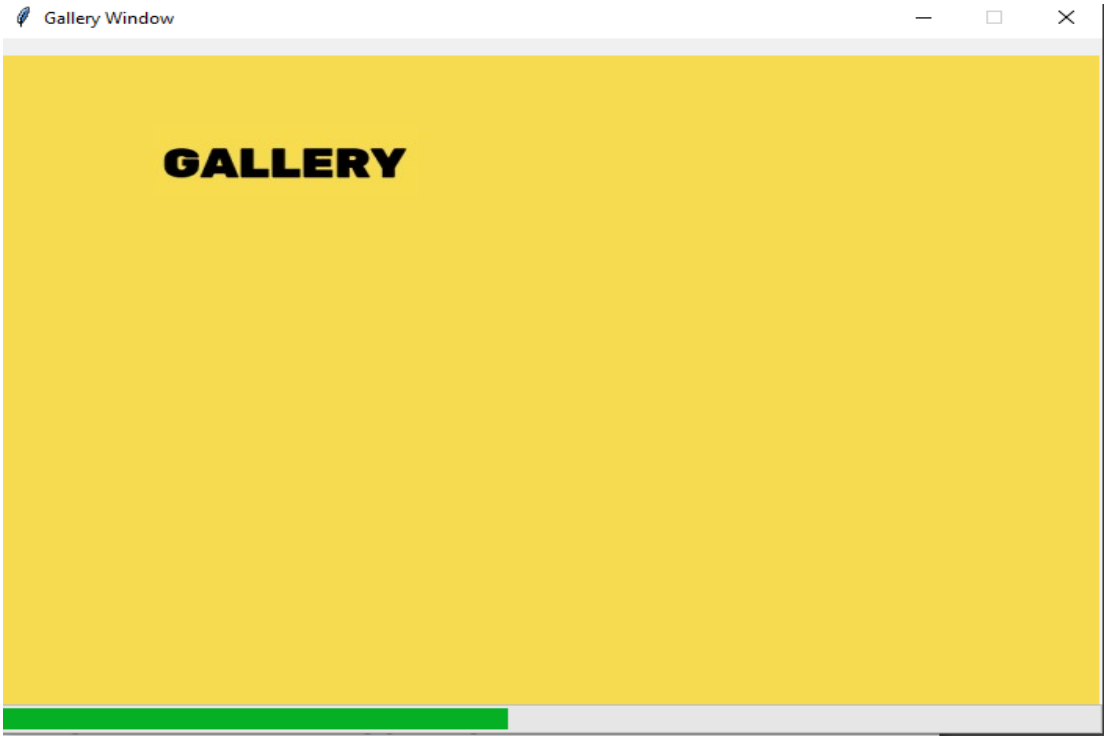


לחיצה על הכפתור פתחה דיאלוג של המשתמש עם File Explorer דיאלוג של הלקוח עם ה File Explorer. על הלקוח לבחור תמונה בה מופיע האדם שהוא מחפש. בנוסף, הדיאלוג מגבל את המשתמש לבחור רק תקיות . ועל כן, הוא לא יכל לבחור קבצים למיניהם.

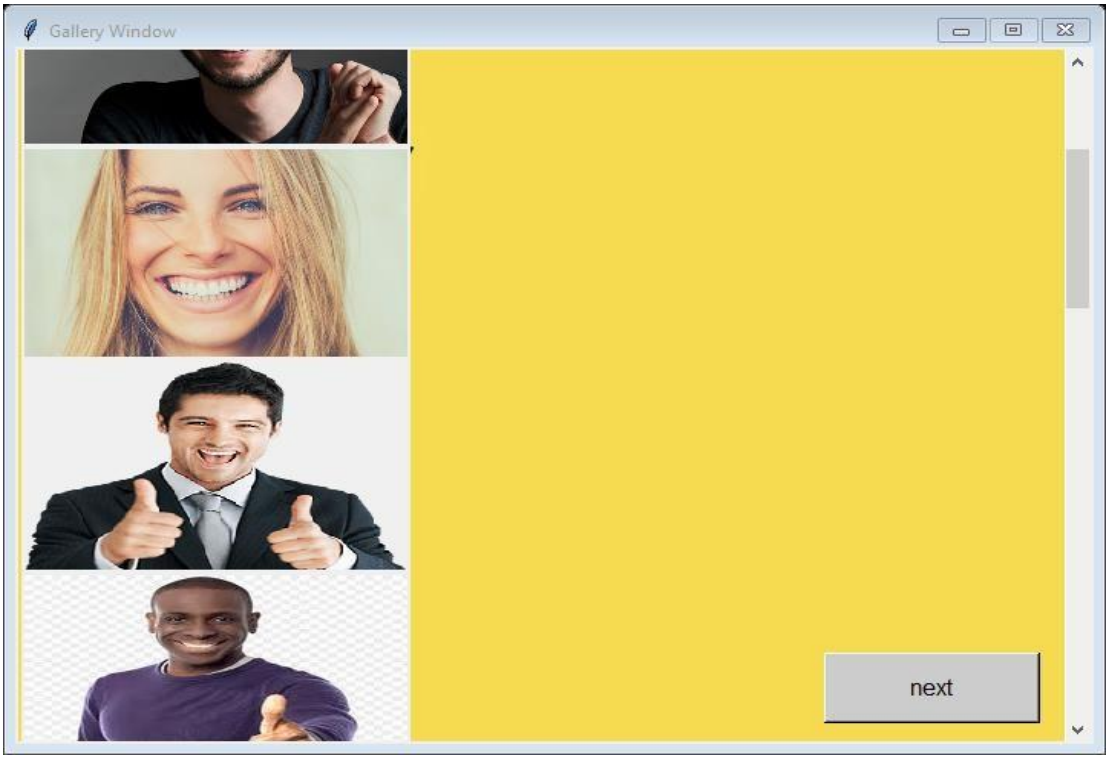
לחיצה על כפתור האיקס וסגירת הדיאלוג תבטל את הלקוח למסך "גלרה")ראה עמוד 34(. המשתמש מעוניין לבחור בגלריית תמונות שנמצאת לו על המחשב

ולכן הוא בחר אותה)בניה
.)images_gallery

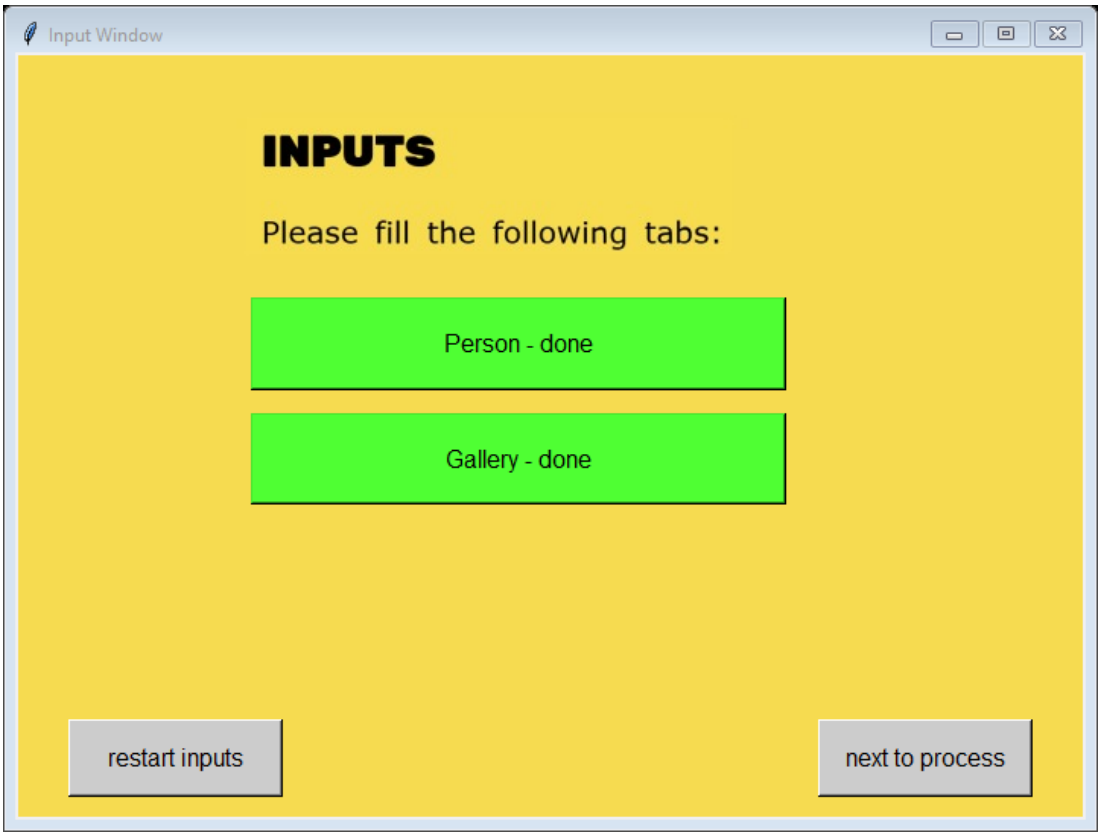
לפני המשתמש נפתח מסך טעינה, שמראה לו עוד כמה בערך זמן נשאר להעלאת התמונות.



בסיום הטענה יופיע למשתמש המסך הבא.



לפני המשתמש נפתח מסך, עם scroll bar, בה יוכל לעיין בכל התמונות בגלריה שהעלה. בנוסף לרשותו כפתור next יעביר אותו לחלון סיום הקלט.



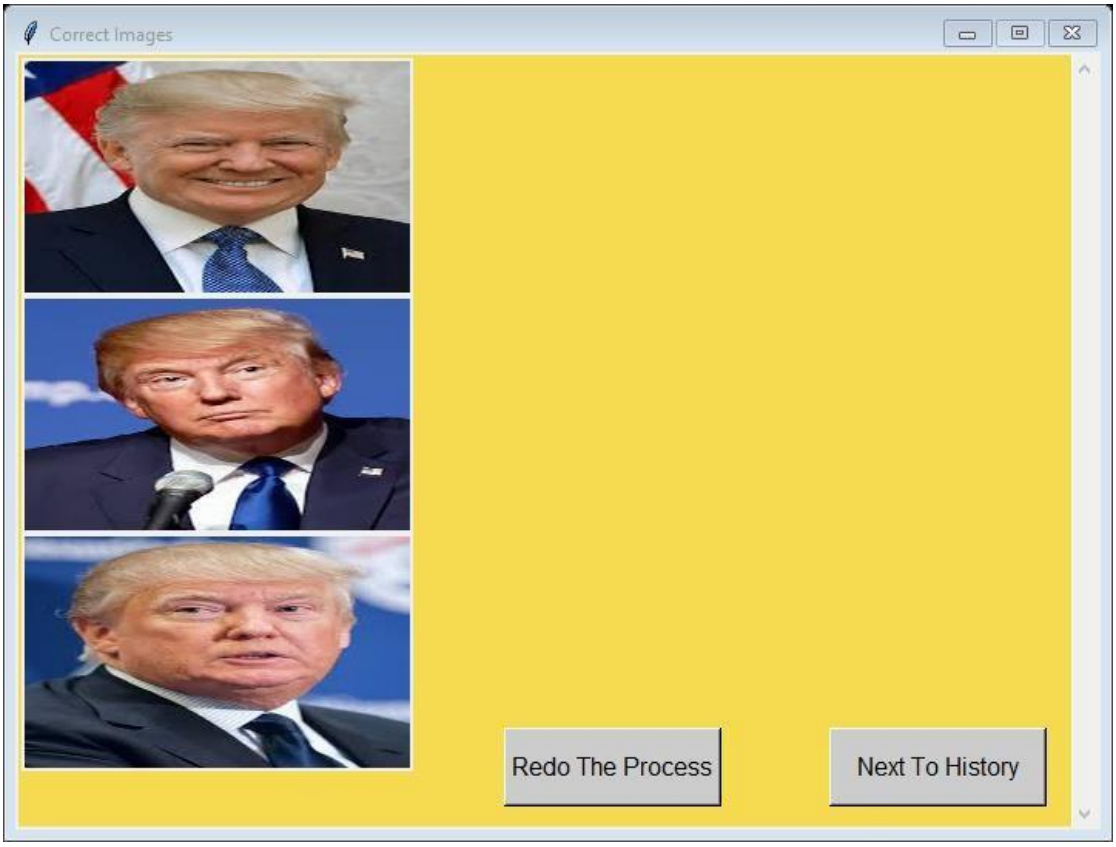
המשתמש סיים ידע שכל הקלט עבר בהצלחה.

לחיצה כל כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש, ותחזיר אותו לסמך הקלט)ראה עמוד 32(.

המשתמש מעוניין להמשיך לתהליך כדי לדעת באילו תמונות שהעלה האדם אותו חיפש נמצא)ניח דונאלד טראמפ(.

בפני המשתמש נפתח מסך טעינה, עד שכל התמונות יעלו.

לאחר סיום הטעינה יפתח מסך ה"תמונות הנכונות".

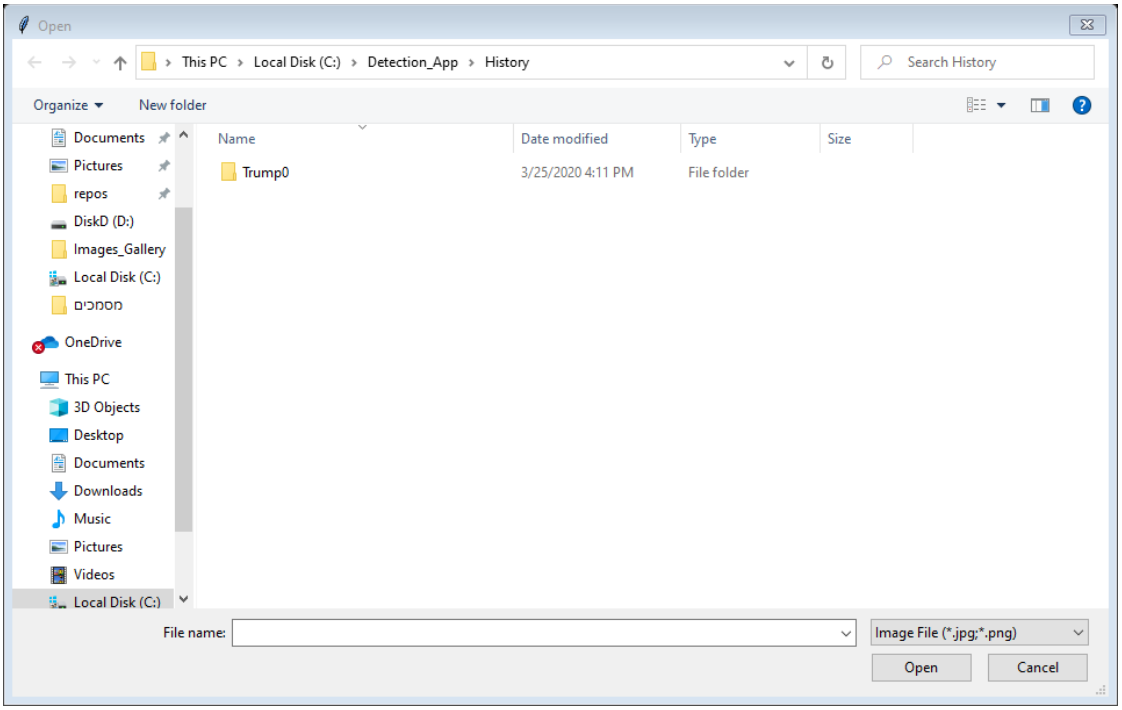


במסך זה למשתמש 2 אפשרויות: ללחוץ על כפתור

redo the process, לחיצה על כפתור זה תוביל את המשתמש למסך הקלט הראשוני)ראה עמוד 32(, שם יתחל את התהליך מחדש. המידע והתהליך הנוכחי שמר לו אצל השרת אבל עוד לא יעלה להסטוריית הלקוח.

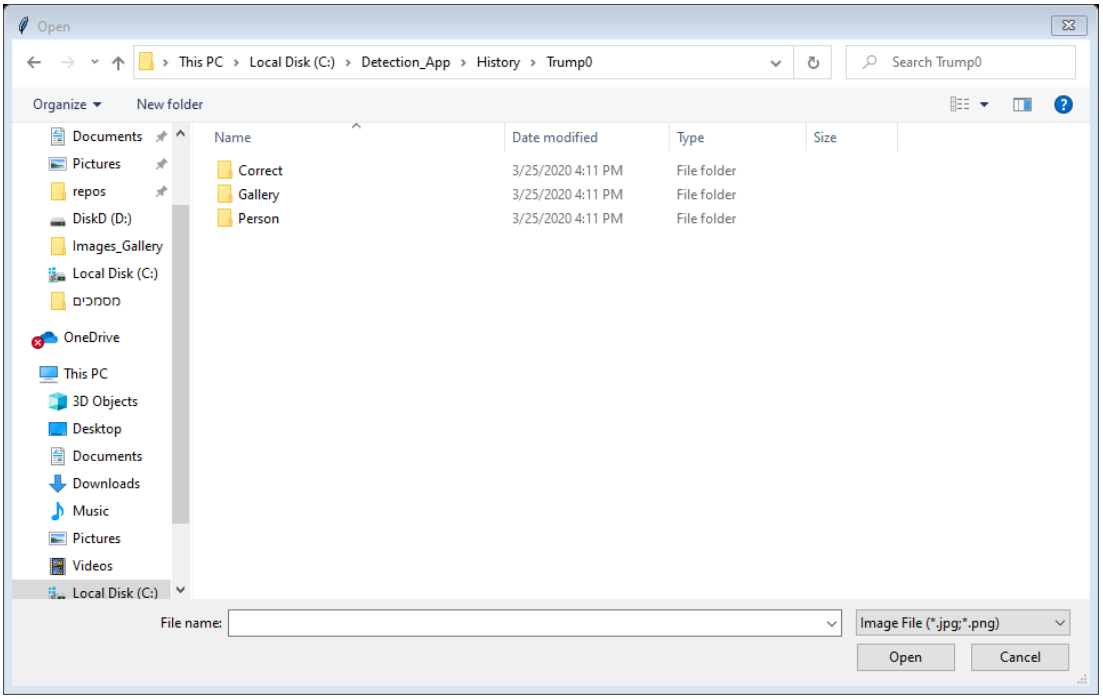
לחיצה על כפתור next to history כשאר המשתמש עשה מספר מסוים של תהלכים וברצונו להסתכל על היסטוריית התהלכים האישית שלו. יפתח מסך טעינה אשר יטען עד סוף העלאת קבצי ההיסטוריה החדשים ללקוח.

אלמוג גל. Face Founder.



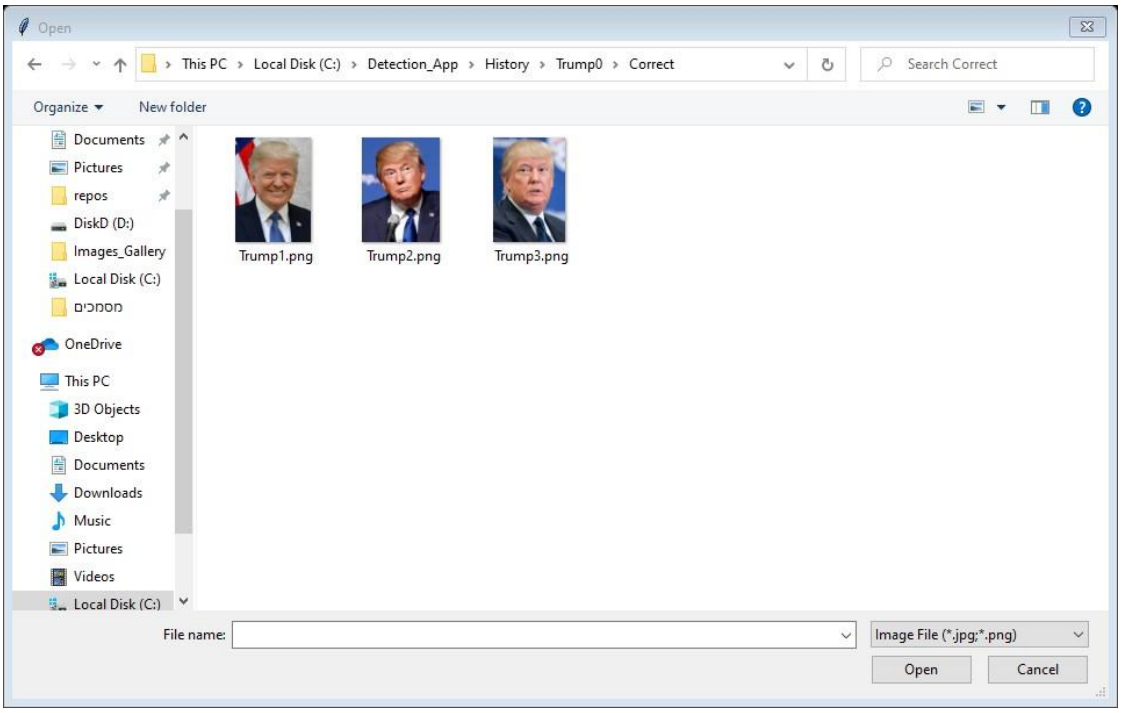
בסיום הטעינה נפתח דיאלוג משולש עם File Explorer. בתיקיה בה ישנם כל החומרים של האפלקציות השונות במחשב ה-windows (כרגע תקיה סתמית). שם יראה הלקוח את כל האנשים אשר חיפש. המשתמש ביצע רק תהליך אחד, וחיפש בפעם הראשונה את דונאלד טראמפ, ועל כן ישנה רק תקיה אחת הנקראת Trump0, 0 כי זהו החיפוש הראשון בוצע על דונאלד טראמפ.

נניח שהמשתמש רוצה עוד מידע על התהליך מסוים שביצע (נניח התהליך הראשון שביצע על דונאלד טראמפ). המשתמש לחץ על התקיה Trump0. לחיצה על כפתור החזרה במעלה העמוד תובל את המשתמש לpath הקודם שעון ב. כך יכל המשתמש לעון ברחבי בגלרה של בחופשות דרך האפלקצה.

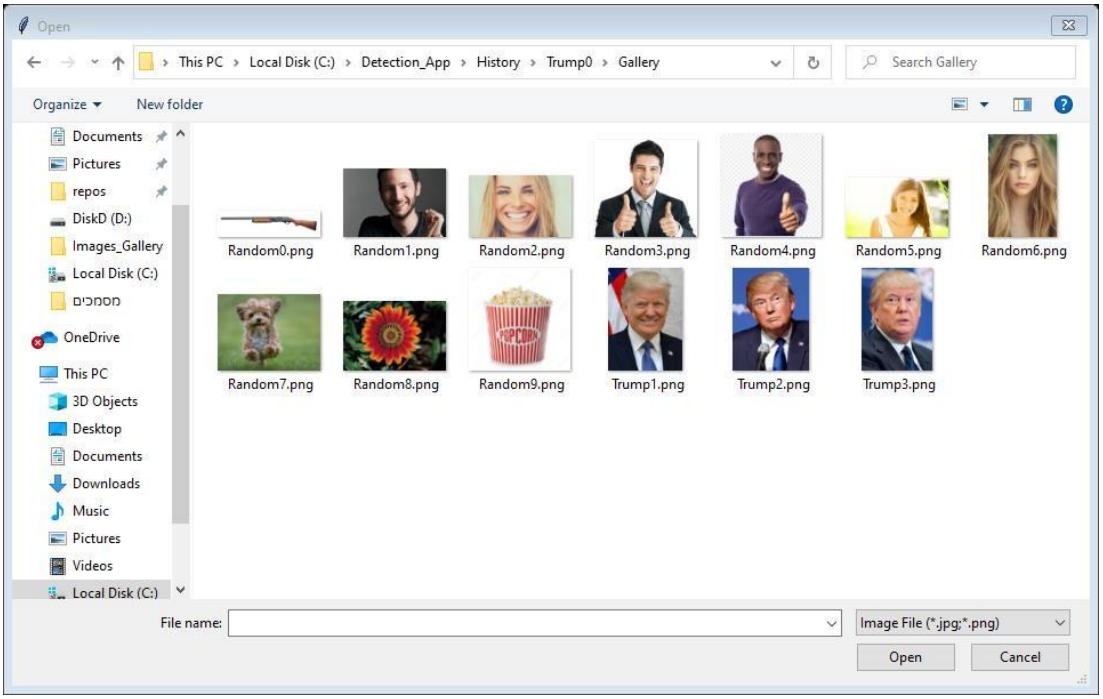


בפני המשתמש מידע על התהליך שביצע ב3 תקיות מסודרות.

Correct – התמונות הנכונות, אל שנמצא בהם האדם המבוקש)ניח דונאלד טראמפ(.

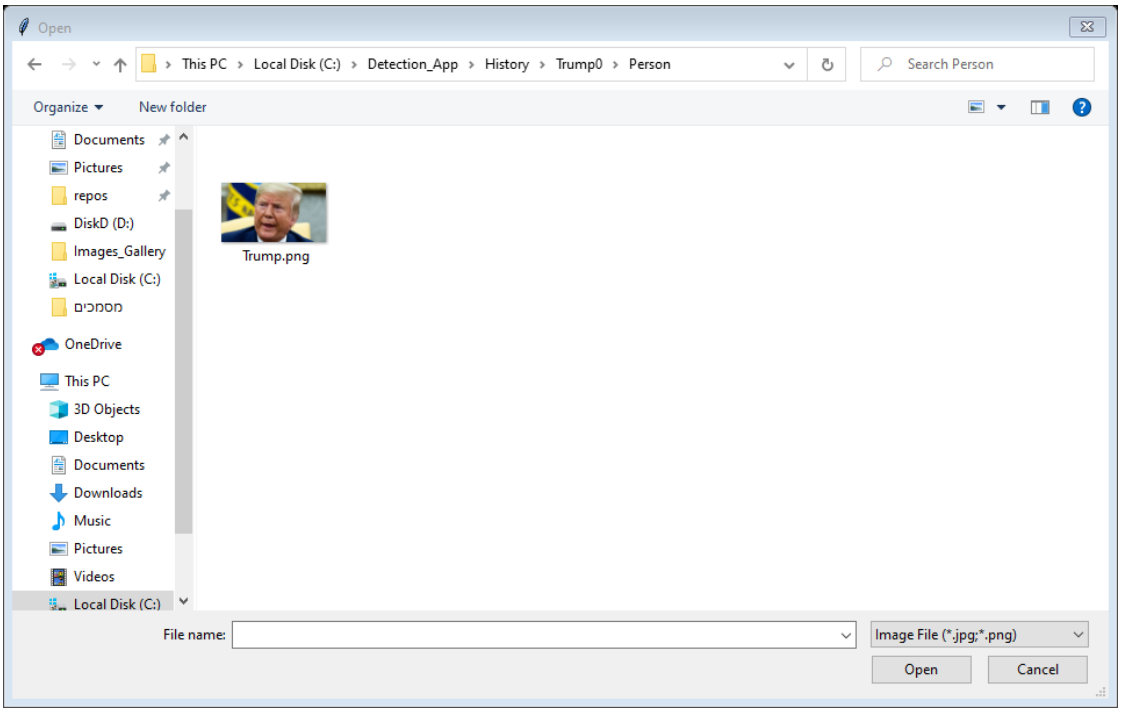


Gallery – גלריית התמונות בהם המשתמש חיפש את האדם המבוקש)ניח דונאלד טראמפ(.



אלמוג גל. Face Founder.

Person – האדם אותו המשתמש רצה לחפש (נניח דונאלד טראמפ).



המשתמש רשאי ללחץ על כפתור האקס שיבל אותו לחלון הסטורה ריק, או ללחץ על תמונה שתבל אותו לחלון הסטורה שבו התמונה שבחר מוצגת. נניח שהמשתמש רצה תמונה מסוימת
(נניח דונאלד טראמפ). הוא לוחץ עליה.

בתחתית העמוד יהה למשתמש 2 אפשרות לעשות את התהליך שב שיחזר אותו לעמוד קלט רק (ראה עמוד 32), או סיום התהליך ויציאה מהאפלקצה שסגור את התכנה את החלון למשתמש
ותבל ליציאה מסודרת. הכפתורים עדיין לא עובב.

מדריך למפתח

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\ Client_Functions.py	לדאוג באופן כלל, שהלקוח פועל באופן תקין, נוחות עם המשתמש ותקשורת רציפה ותקינה עם השרת.	כל הפונקציות המרכזיות של הלקוח. הפעלה וסנכרון פונקציות אלו על פי מעשי המשתמש, תוך כדי הצגת ממשק גרפי. שליחת הודעות שרת בהתאם. יצירת מפתח סימטרי פרטי עם השרת.	General Client	Client_Functions.py

תדפיס הקוד:

הכנה של משתנים גלובליים: paths של רקעים, גודל ראשוני של המסך, ערכים ראשוניים למשתנים למיניהם בתוכנית, סוגי הקבצים שהמשתמש רשאי לעלות, events, רקעים, paths של תיקיות היסטוריה ותיקיות זמנות במחשב הלקוח.

משתנים קבועים: בכל הפונקציות קיים משתנה username – נועד כדי להזדהה בשליחת ההודעות לשרת.

בקשה לקבלת מפתח ציבורי מהשרת. בקשה ליצירת מפתח

סימטרי פרטי מהשרת, יצירת מפתח סימטרי פרטי ושמירתו אצל הלקוח.

מחיקת כל הווידג'טים הקודמים, פתיחת החלון הפתיחה, יצירת רקע, שני כפתורים אפורים: כניסה למערכת- ובל להכנסת פרטי חשבון קיים ומשתמש חדש- הכנסת פרטים של חשבון חדש, כותרת, event של יציאה של המשתמש באופן לא מסודר. **משתנים:** background – path של רקע המסך בהתאם למצב שבו נמצא המשתמש.

מחיקת כל הווידג'טים הקודמים, החלפת החלון של משתמש חדש. יצירת רקע, כותרת – "חשבון חדש", שני תאי מילוי אפורים: אחד רשום בתוכו "פה הכנס שם משתמש" שנמחק ברגע שהמשתמש נמצא על התא ושם המשתמש יכניס את שם המשתמש ושני רשום בתוכו "פה הכנס את הסיסמא" שנמחק ברגע שהמשתמש לו נמצא על התא. שני events אחד של חץ למעלה ואחד של חץ למטה שמאפשרים למע, בנוסף על ללחיצות, בין התאים. Event נוסף של לחצן הenter שפעיל את כפתור האישור. כפתור "חזור" שיחזיר את המשתמש

למסך הפתיחה. כפתור "אישור" שישלח אל השרת את הפרטים של ניסון ההתחברות לחשבון חדש. במידה ושנה איזושהי תקלה השרת ישלח ללקוח הודעת שגיאה ובה ירשום

את התקלה. השרת יקרא לפונקציה הזאתי ושנה את הרקע לרקע שגיאה בהתאם לטעות. במקרה של הכנסת פרטים נכונה, הלקוח יחזור למסך הפתיחה שם ישונה הרקע לרקע שבו יש הודעה על הצלחת פתיחת החשבון החדש. כפתור החבא אשר יסתיר את הסיסמא, יקרא לפונקציה החבא סיסמא. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** background – path של רקע המסך בהתאם למצב שבו נמצא המשתמש, username entry

אלמוג גל. Face Founder.

— מה היה בתא המילוי של שם המשתמש, במידה ויש קריאה חוזרת לפונקציה, entry
password— מה היה בתא המילוי של שם הסיסמא, במידה ויש קריאה חוזרת לפונקציה.

מחיקת כל הווידג'טים הקודמים, החלפת חלון כניסה לחשבון קיים. יצירת רקע, כותרת — "כניסה", שני תאי מילוי אפורים: אחד רשום בתוכו "פה הכנס שם משתמש" שנמחק ברגע שהמשתמש נמצא על התא ושם המשתמש יכניס את שם המשתמש ושני רשום בתוכו "פה הכנס את הסיסמא" שנמחק ברגע שהמשתמש לוחץ נמצא על התא. שני events אחד של חץ למעלה ואחד של חץ למטה שמאפשרים לנוע, בנוסף על ללחיצות, בין התאים.
Event נוסף של לחצן הenter שיפעיל את כפתור האישור. כפתור "חזור" שיחזיר את המשתמש למסך הפתיחה. כפתור "אישור" ששלח אל השרת את הפרטים של ניסיון ההתחברות לחשבון
קיים. במידה ושנה איזושהי תקלה השרת ישלח ללקוח הודעת שגיאה ובה ירשום את
התקלה. השרת יקרא לפונקציה הזאתי ושנה את הרקע לרקע שגיאה בהתאם לטעות. במקרה של הכנסת פרטים נכונה, הלקוח יעבור לחלון הוראות,
כניסתו לחשבון בוצעה
בהצלחה. כפתור החבא אשר יסתיר את הסיסמא, יקרא לפונקציה החבא סיסמא. event של יציאה של המשתמש באופן לא מסודר. **משתנים:**
path — background של רקע המסך בהתאם למצב שבו נמצא המשתמש, username entry — מה היה בתא המילוי
של שם
המשתמש, במידה ויש קריאה חוזרת לפונקציה, password entry— מה היה בתא המילוי של
שם הסיסמא, במידה ויש קריאה חוזרת לפונקציה.

מחיקת כל הווידג'טים הקודמים, החלפת רקע וחלון לרקע וחלון הוראות, כותרת — "איך משתמשים", event של לחיצה על enter לעבור לחלון
הקלט הראשוני, ויצירת כפתור הבא שלחיצה עליו גם תעבור לחלון הקלט הראשוני. event של יציאה של המשתמש באופן לא
מסודר. **משתנים:** path — background של רקע המסך בהתאם למצב שבו נמצא המשתמש.

מחיקת כל הווידג'טים הקודמים, החלפת חלון לחלון הקלט, החלפת רקע לרקע הקלט, כותרת — "חלון קלט", כפתור איפוס קלט אשר מאפס את כל הקלטים שהוכנסו בשלון הזה וקורא לפונקציה מחדש)בנוסף, השרת מוחק תהליכים שלא הושלמו ולכן קיימת פה מחיקת תהליך(. איפוס event על לחיצת enter ללא לעשות כלום. אם עוד לא הוכנס קלט של אדם יוצג כפתור שרשום עליו אדם, שלחיצה עליו תעבור למסך קליטת אדם. אם הוכנס קלט של אדם, אז לא הוכנס קלט של גלריה ולכן תוצג הודעה שהאדם נקלט בהצלחה, יחד עם כפתור הכנס גלריה שלחיצה עליו תעבור לחלון קלט הגלריה. אם הוכנס גלריה, סימן שגם הוכנס
אדם, ולכן 2 הקלטים סומנו בהצלחה, יוצג הודעה שגם האדם וגם הגלריה נקלטו בהצלחה, ויוצג כפתור המשך לתהליך שלחיצה עליו תוביל לתהליך הפנים.
event של יציאה של
המשתמש באופן לא מסודר. **משתנים:** path — background של רקע המסך בהתאם למצב שבו נמצא המשתמש, אדם- בוליאני,
האם בוצע קלט של אדם או לא, גלריה- האם בוצע קלט
של גלריה או לא)המעיד על סוג הקלט, קיים סדר כרונולוגי בין האדם לגלריה(.

מחיקת כל הווידג'טים הקודמים, החלפת חלון לחלון הקלט של אדם, החלפת הרקע לרקע קלט של אדם, כותרת — "חלון אדם", יוצג כפתור של בחירת תמונה שלחיצה עליו תוביל לפונקציית דיאלוג בין המשתמש לסייר קבצים. אם קיים איזשהו path סימן שהמשתמש כבר בחר תמונה תקינה, על כן הלקוח ישלח בקשה לשרת ושאל אותו האם בתמונה הזאת יש פרצוף אחד. אם אין פנים שווה 2 סימן שיש יותר מפנים אחד בתמונה, על פי השרת, ויקראו לפונקציה הזאת שוב עם רקע שממנו יוצג טעות שלפיה יש יותר מפרצוף אחד. אם פנים שווה לO סימן שאין פנים בתמונה, על פי השרת, ויקראו לפונקציה הזאת שוב עם רקע שממנו יוצג טעות שלפיה יש אין פנים. אם פנים שווה ל1 סימן שבתמונה יש אדם אחד, והיא תקינה לחלוטין. הלקוח יבחר את שם התיקייה בה ישמור הלקוח את הפרטים לתמונה הנ"ל לפי תא מילוי. לאחר המילוי ישלח שם התיקייה והמידע הבינארי של התמונה הנ"ל לשרת

אלמוג גל. Face Founder.

בבקשה לפתוח תיקיית היסטוריה תחת השם הזה, לתמונה הנ"ל. השרת יאשר. הלקוח יוצה להצג למשתמש את התמונה אשר בחר. לכן יפתח path זמני שם ישמור את התמונה, ישנה את גודלה לגודל שיתאים למסך, ישמור אותה כpng)בtkinter ניתן להציג רק תמונות מסוג png(, ויציג אותה.
לאחר הצגת התמונה יופיע למשתמש שתי כפתורים: כפתור בחר תמונה אחרת שתקרא לפונקציה הנ"ל מהתחלה שוב)והשרת ימחק את התהליך הנ"ל, כי עוד לא

השלם(וכפתור הבא שלחיצה עליו תקרא לחלון הקלט עם אדם. event של יציאה של המשתמש באופן לא מסודר. **משתנים:**
path – background של רקע המסך בהתאם למצב
שבו נמצא המשתמש, path – תמונה של האדם.

מחיקת כל הווידגטים הקודמים, החלפת החלון לחלון קלט של גלריה, החלפת הרקע לרקע של קלט של גלריה, כותרת- "חלון גלריה", אם אין path סימן שהמשתמש לא בחר עדיין
גלריה, לכן יקראו לפונקציית הדיאלוג יחד עם גלריה. אם כן path סימן שהמשתמש בחר כבר גלריה. ישנו תא מילוי שבו המשתמש מתבקש להכניס את שם הגלריה. לאחר מכן נשלחת
בקשה מהלקוח לשרת לפתוח תיקיית גלריה יחד עם תיקיית היסטוריה האחרונה שהייתה, בשם שהלקוח בחר. קוראים לפונקציה מצא את סוגי הקבצים שמחזירה מתוך path מסוים רק את הקבצים מסוג מסוים. לאחר מכן הפונקציה עוברת אחרי כל קובץ, בלולאה, ומבקשת
מהשרת להכניס את כל קובץ לגלריה אצל תיקיית ההיסטוריה העכשווית ביותר שהייתה. לאחר מכן הפונקציה יוצרת מד טעינה שעולה בחלוקה שווה בין מספר התמונות, כשאר בכל פעם שהיא מקבלת הודעה מהשרת שהתמונה הוכנסה לגלריה בהצלחה המד עולה. לאחר
מכן, בין העלאת התמונות, עדיין בלולאה הלקוח במקביל שומר את התמונות על המחשב של הלקוח, באופן זמני, בpath זמני ומשנה את גודלן, ברגע שכל התמונות הועלו לשרת ונשמרו אצל הלקוח, מד הטעינה נהרס. קוראים לעצם "גלריה מתגלגלת" יחד עם הpath הזמני שבו נמצאות התמונות גלריה ששמורות במחשב של המשתמש.)ראה הסבר על העצם הזה
בהמשך(. התמונות מוצגות יחד עם פס גילה אנכי, ומאפסים את הרשימה שבה היו התמונות גלריה מהסוג הנכון, לפעם הבאה כי זהו משתנה גלובלי. כאשר מוצגות כל התמונות גלריה יופיע גם כפתור "הבא" שלחיצה עליו תובל לחלון הקלט עם גלריה. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך
בהתאם למצב שבו נמצא המשתמש, path – הpath של גלריית התמונות.

פונקציית מצא את סוגי הקבצים השונים עוברת על כל התיקיות בpath ומתוכן מוציאה תתי תיקיות, בצורה רקורסיבית, עד שמגיעה לכל הקבצים מהpath המקורי. כל פעם שמגיעה לקובץ, בודקת האם הוא קובץ מהסוג הרצוי. אם כן, מכניסה אותו לרשימה, אם לא מתעלמת. הפונקציה מחזירה את הרשימה עם הקבצים מהסוג הנכון. **משתנים:** path –
הpath בוא יש למצוא את סוגי הקבצים הנכונים, types – הסוגים אשר צריך למצוא.

פונקציית הדיאלוג בין המשתמש לסייר הקבצים בודקת האם קיים דיאלוג כי המשתמש בחלון האדם או בחלון הגלריה או בחלון ההיסטוריה. אם בחלון האדם, היא מאפשרת למשתמש לבחור רק סוגי קבצים לפי משתנה גלובלי של סוגי קבצים קבועים, ולאחר בחירת path תקין קוראת חזרה לפונקציית האדם עם יחד עם הpath. אם גלריה, הפונקציה מאפשרת
למשתמש לבחור רק תיקיה, וברגע שבחר directory תקין קוראת חזרה לפונקציית הגלריה
יחד עם הpath. אם היסטוריה, הפונקציה מאפשרת למשתמש לבחור רק קבצים מסוג
מותר, לפי משתנה גלובלי, והdirectory הראשוני שמוצג לאדם הוא הdirectory של ההיסטוריה האישית במחשב של הלקוח.
משתנים: person – בוליאני, האם מדובר בחלון
קלט של אדם או לא, history – בוליאני, האם מדובר בחלון היסטוריה או לא, gallery -
בוליאני, האם מדובר בחלון גלריה או לא.

אלמוג גל. Face Founder.

מחיקת כל הווידג'טים הקודמים, יצירת רקע של תהליך הפנים, כותרת- "תמונות נכונות", אם אין עוד רשימה של התמונות הנכונות, סימן שהלקוח עדיין לא התחיל את התהליך, ולכן
ישלח הלקוח בקשה לשרת שישלח לו את כל התמונות הנכונות — בתיקייה הכי עכשווית,
להשוות את התמונה של האדם לכל תמונות הגלריה, ולמצוא באילו תמונות האדם נמצא, ולהחזיר ללקוח. השרת ישלח חזרה ללקוח תמונות בהם נמצא
האדם. הלקוח יקרא לפונקציית הכנסת תמונות נכונות, עם שם התמונה, התמונה, ושם המשתמש. בזמן הזה יקראו לפונקציית ההמתנה. אם יש תמונות
נכונות, סימן שהתהליך כבר בוצע ע"י השרת
והתמונות כבר בידי הלקוח. לכן השרת יעבור על כל תמונה ישמור אותה בpath זמני, ישנה את גודלה. יהיה למשתמש מד טעינה עד שהתמונות יעלו. הלקוח
יקרא לעצם "גלריה מתגלגלת" יחד עם הpath הזמני שבו נמצאות התמונות גלריה ששמורות במחשב של
המשתמש.)ראה הסבר על העצם הזה בהמשך(. התמונות מוצגות יחד עם פס גלילה אנכי, ומאפסים את הרשימה שבה היו התמונות גלריה מהסוג הנכון,
לפעם הבאה כי זהו משתנה גלובלי. בסוף התהליך יהיה ללקוח שתי אפשרויות: כפתור עשה את התהליך שוב, שיקרא שוב לחלון הקלט)התהליך הנוכחי
הושלם ולכן כן ישמר ע"י השרת(וכפתור המשך
להיסטוריה שיקרא לפונקציית היסטוריה. event של יציאה של המשתמש באופן לא מסודר.
משתנים: path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש,
correct images – הpaths של תמונות הנכונות בהם נמצא האדם המבוקש.

מחיקת כל הווידג'טים הקודמים, החלפת הרקע לרקע של "המתן בבקשה", כותרת "חלון המתנה", בסוף התהליך תקרא, תקרא לפונקציה שקבלת
כקלט. event של יציאה של
המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב
שבו נמצא המשתמש,

פונקציית הכנס תמונות נכונות מכניסה לרשימה גלובלית של תמונות נכונות כל תמונה שהשרת טוען שבה נמצא האדם, זאת באמצעות יצירת התמונה
בpath זמני, שמירתה)יחד עם התאריך והשעה המדויקים, למנוע כפל שמות(, ולאחר מכן שולח הודעה שהתמונה
הוכנסה בהצלחה, על מנת לחכות לתמונה הבאה. שם המשתמש לזיהוי בפני השרת
בהודעות. **משתנים:** name – שם התמונה, data – מידע בינארי של התמונה.

מחיקת כל הווידג'טים הקודמים, שינוי הרקע לרקע היסטוריה, כותרת- "היסטוריה", אם לא סיים סימן שעוד לא הועלתה ההיסטוריה, אם גם path ההיסטוריה
לא קיים, זהו ההיסטוריה הראשונה שמועלת למשתמש לכן תיצור path קבוע של היסטוריית המשתמש, אם הpath קיים תשאיר את המצב כמו שהוא.
אין לך תיקות היסטוריה)עדיין לא סיים(, תשלח בקשה לשרת שיעלה את השמות של תיקיות ההיסטוריה. לאחר מכן תקרא לפונקציה הזאת אם
התיקיות. אם כן תיקות היסטוריה, תפתח בpath הקבוע תיקות לפי השמות ששלח השרת. לסוף תשלח בקשה לשרת שיתחיל לעלות קבצי היסטוריה. בכל
פעם שמגיע קובץ היסטוריה הוא מועבר לפונקציה הכנס תיקות היסטוריה, ששומרת את הקבצים. בסוף העלאת כל
הקבצים השרת יודע שאין יותר היסטוריה חדשה, ואז הלקוח יקרא לפונקציה הזו שכן סיים. אם סיים תקרא לפונקציית הדיאלוג עם היסטוריה. אם המשתמש בוחר
איזשהו קובץ, הלקוח קורא לפונקציה הזו שכן סיים ועם path של התמונה. הפונקציה תשמור את התמונה הpath זמני, תשנה את גודלה, את סוגה לpng
)tkinter מציג תמונות מסוג png(, ותציג את
התמונה. בסוף החלון יהיה למשתמש האפשרות לראות שוב את ההיסטוריה שלו, שיקרא לפונקציה הזו עם סיים, לעשות את התהליך שוב, שתקרא לפונקציית
הקלט או לצאת
מהאפליקציה מה שיובל ליציאה מסודרת. event של יציאה של המשתמש באופן לא מסודר.
משתנים: path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש,
history folders – השמות של תיקות ההיסטוריה, image path – תמונה אשר המשתמש
רוצה לראות, done – בוליאני, האם המשתמש סיים את העלאת ההיסטוריה או לא.

אלמוג גל. Face Founder.

פונקציית יצירת מפתח סימטרי, שיוצרת על פי אלגוריתם הRSA מפתח סימטרי פרטי, יחד
עם השרת. **משתנים:** min value – ערך מינימלי,max value – ערך מקסימלי.

הוספת אפסים- מוסיפה אפסים לגודל ההודעה, כך שהשרת ידע מה גודל ההודעה, בנוסף
מודיעה ללקוח אם ההודעה גדולה מידי. **משתנים:** str1 – איזשהי מחרוזת.

טיפול הודעות דואגת שהודעות גדולות מדי)מידע של קובץ תמונה(לא יודפסו אלה יודפס
במקום מידע של קובץ תמונה. **משתנים:** message – ההודעה.

פונקציית בדיקה בודקת את הפאקטות שקיבל הלקוח ופועלת בהתאם. תחילה מדפיסה את טיפול ההודעות של ההודעה. לאחר מכן היא בודקת אם במקום הראשון בפאקטה יש אישור
או טעות. אם אישור: יציאה- ניתוק מהתוכנית, מפתח ציבורי נשלח- שמירת המפתח כגלובל,
משתמש חדש נוצר- תקרא לחלון הפתיחה עם רקע עם הודעה שהחלון נוצר בהצלחה, כניסה למערכת- תקרא לחלון ההוראות, אדם בתמונה הזו –
תקרא לפונקציה הכנס תמונות נכנות, אם אין יותר תמונות נכנות- תקרא לתהליך פנים עם התמונות הנכנות שנשלחו, אם הגה תיקיות ההיסטוריה- תקרא להיסטוריה עם
התיקיות הנ"ל, אם תכניס את הקובץ
להיסטוריה- תקרא להכנסת קבצים להיסטוריה, אם אין עוד קבצי היסטוריה – תקרא למסך ההיסטוריה עם הקבצים שנשלחו עד כה. אם טעות: אין שם
משתמש- תקרא למסך משתמש חדש עם רקע שמודיע שלא הכנס שם משתמש, אם אין סיסמא- תקרא לפונקציה משתמש חדש עם רקע שמודיע שלא
הכנסה סיסמא, אם שם משתמש קצר מידי- תקרא לפונקציה
משתמש חדש עם רקע שמודיע ששם המשתמש קצר מידי, אם סיסמא קצרה מידי- תקרא למשתמש חדש עם רקע שמודיע שהסיסמא קצרה מידי,
אם שם משתמש קיים- תקרא למשתמש חדש יחד עם רגע שמודיע ששם המשתמש הזה כבר תפוס, אם שם המשתמש לא תואם לסיסמא-
תקרא לכניסה לחשבון עם רקע שמודיע שהשם המשתמש או הסיסמא שגויים, אם המשתמש כרגע בשימוש- תקרא לכניסה לחשבון יחד עם רקע שאומר
שאי אשפר להיכנס לאותו חשבון במקביל. **משתנים:** data – ההודעה, username – שם
המשתמש, password – הסיסמא של המשתמש.

פונקציית שליחה וקבלה. אם מפתח סימטרי פרטי תצפן בעזרת המפתח הסימטרי הפרטי את ההודעות שאתה שולח לשרת. אם לא מפתח סימטרי פרטי,
תצפן בעזרת RSA את ההודעה שאתה שולח כדי להחזיק יחד עם השרת מפתח ציבורי פרטי. תקרא להוספת
אפסים לגודל הפאקטה שברצונך לשלוח, ומצרפת את הפאקטה המוצפנת עם ההודעה, שעוברת הפיכה למחרוזת)pickle(. זאת הפאקטה החדשה.
תשלח את הפאקטה: גודל והודעה מוצפנת לשרת. אם שלחת פאקטה שגודלה גדול מ3 ז"א שאתה רוצה לקבל תשובה חזרה, ובעצם לחכות\להיתקע בקבלת
הודעה. כאשר אתה מקבל את ההודעה תבדוק מה
הגודל שלה, תחזיר אותה לרשימה)pickle(, תפענח אותה בעזרת המפתח ותקרא
לפונקציית הבדיקה יחד עם ההודעה. **משתנים:** data – ההודעה, use symmetrical key –
בוליאני, האם להשתמש במפתח הסימטרי הפרטי או לא)אם הוא קיים או לא(

פונקציית החבא סיסמא משנה את כל התווים בתא המילון של הסיסמא לכוכביות, ובמידה אם הם כבר כוכביות משנה אותם בחזרה. **משתנים:** תא
מילון- תא המילון של הסיסמא עליו
יבוצעו הפעולות. כפתור החבא אשר יסתיר את הסיסמא.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\ Communication_Settings.py	לדאוג באופן כלל, שהלקוח פועל באופן תקין, נוחות המשתמש ותקשורת עם רציפה ותקינה עם השרת.	קובץ כפול בצד השרת והלקוח שנועד להבטיח שהלקוח יוכל לתקשר עם השרת.	General Client	Communication_Settings.py

ההגדרות הכפולות של התקשרות. **משתנים:**

```
Host = '127.0.0.1'  #
Port = 9998 # the port. important that the clients
and the server be in the same port
Max_Data_Size = 9 # the biggest size of data that
can be sent. 999,999,999 bits.
```

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\ Handle_Images.py	לדאוג באופן כלל, שהלקוח פועל באופן תקין, נוחות המשתמש ותקשורת עם רציפה ותקינה עם השרת.	קבצים מרכזיתנפרדת מקובץ הפונקציות המרכזיות של השרת, שדואגת להציג הרבה תמונות, עם מד טעינה ופס גלילה.	General Client	Handle_Images.py

שני פונקציות שדואגות לטיפול בתמונה בGUI:

עצם להצגת גלריה, מקבלת path שנמצא בו תמונות גלריה, מציגה אותם זו אחר זו, ומוסיפה פס גלילה אוני כדי לעין בתמונות. רקע- כרקע האפליקציה.

משתנים: parent – החלון

ה"אב", תמונות.

עצם להצגת תמונה יחידה בגדול, רקע- כרקע האפליקציה. **משתנים:** parent – החלון

ה"אב", תמונות.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\ Encryption_And_Decryption.py	לדאוג שמידע אישי ורגיש של הלקוח לא יחשף	קובץ כפול הדואג להצפין ולפענח הודעות לפי מפתח ואלגוריתם	Security Client	Encryption_And_Decryption.py

	בעת תקשורת עם השרת.	זהה בני הצדדים, וכך להבטיח אבטחה הבנה של ההודעות בתקשורת.		

כאשר לקוח מעוניין להתחבר הוא מקבל את המפתח הציבורי. לאחר מכן רק לקוח שיודע את הפרוטוקול של השרת)אם כי עדיין בעל סיכוי לניחוש\פריצה(יודע שהוא צריך לייצר בעצמו עוד מספר. יצירת המספר של הלקוח היא יצירת מספר רנדומאלי)באמצעות Random(בין המספר הציבורי השני שקיבל לבין מספר היכולת הרצה של המחשב שלך)ככל שהכוח המעבד שלך קטן יותר כך המספר יקטן(. מכיוון שאני עובד במחשב ביתי, ואין ברשותי כוח מעבד כמו חברות אבטחה גדולות שמגיעות גם למספרים של 50 ו60 ספרות, מספר

ההגבלה שלי הוא 150000. הגעתי למספר זה באמצעות לולאה ששיניתי כל פעם את המספר עד שהגעתי למספר ההגבלה הגדול ביותר וזמן ההרצה הקטן ביותר)לא יותר מכמה שניות במקרה הגרוע ביותר(. לאחר יצירת המספר הנ"ל הלקוח מעלה את המספר שצר בחזקת המספר הראשון הציבורי והשארית של התוצאה מהמספר השני הציבורי תיתן את המפתח הסימטרי הפרטי. הלקוח שולח את המספר הרנדומלי)לא הפרטי(שמצא לשרת.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה הוא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הוא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות
ביניהן כך שרק הם יודעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה)שאפרט בהמשך(ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין
ברשת, בלי המספר הפרטי של השרת d הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של
הלקוח והשרת.

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח
והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים יודע מה המידע בהודעה. אלגוריתם ההצפנה בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה)הודעה(מסוג
רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו)באמצעות פונקציית ord(. את המספר הנ"ל הופכים)נגד 1546 הופך
ל6451(. מחסירים ממנו 100 עד שנמצא בין 0 ל255, בינתיים שנו מונה)ערכו הראשוני 40(שעולה ,בכל החסרה של המספר ההפוך, במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה
למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג

התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח יהיה מסוגל להעביר את האותיות הascii למקור, שכן הוא יודע כמה)בעזרת ציקוף המונה ליד כל אות(פעמים להכפל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל
ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק

אלמוג גל. Face Founder.

מידע מוצפן. גם בעזרת brute force לא ימצא מהי הפאקטה המקורית כי אין ברשותו את המפתח. כאמור הוא יכל לנחש פעם את המפתח ועלי לנסות brute force, אבל כמובן דבר זה ייקח לו שנים. חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי מאלפי תווים אבל זהו בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהא רשימה, עוברת pickle, שהופך אותה לstring ארוך. דבר שמקשה עוד יותר על גורם שלישי עין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם רק השרת והלקוח
יודעים את ההודעות שהם שולחים זה לזה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ blank.jpg	עיצוב גרפי של האפליקציה.	רקע ריק, למקרה הצורך.	Backgrounds	blank.jpg

רקע ריק, למקרה הצורך.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ detection_face.jpg	עיצוב גרפי של האפליקציה.	רקע בו יוצג האדם אותו רוצים לחפש.	Backgrounds	detection_face.jpg

רקע בו יוצג האדם אותו רוצים לחפש.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ exists_user.jpg	עיצוב גרפי של האפליקציה.	רקע להכנסת הפרטים למיסה לחשבון קיים.	Backgrounds	exists_user.jpg

רקע להכנסת הפרטים למיסה לחשבון קיים.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ gallery.jpg	עיצוב גרפי של האפליקציה.	רקע להצגת גלריית התמונות מתוכו חיפש המשתמש אדם מסוים.	Backgrounds	gallery.jpg

רקע להצגת גלריית התמונות מתוכו חיפש המשתמש אדם מסוים.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ history.jpg	עיצוב גרפי של האפליקציה.	רקע להצגתההיסטוריה של המשתמש.	Backgrounds	history.jpg

רקע להצגת ההיסטוריה של המשתמש.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ How_to_use.jpg	עיצוב גרפי של האפליקציה.	רקע וחלון המסביר כיצד להשתמש ומה השימוש של האפליקציה.	Backgrounds	How_to_use.jpg

רקע וחלון המסביר כיצד להשתמש ומה השימוש של האפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ inputs.jpg	עיצוב גרפי של האפליקציה.	רקע לחלון קלט הנתונים של המשתמש.	Backgrounds	inputs.jpg

רקע לחלון קלט הנתונים של המשתמש.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ Main.jpg	עיצוב גרפי של האפליקציה.	רקע הפתיחה של האפליקציה.	Backgrounds	Main.jpg

רקע הפתיחה של האפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds \new_user.jpg	עיצוב גרפי של האפליקציה.	רקע להרשמתמשתמש חדש לאפליקציה.	Backgrounds	new_user.jpg

רקע להרשמת משתמש חדש לאפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\ register_user_main.jpg	עיצוב גרפי של האפליקציה.	רקע הפתיחה עם הודע שההתחברות של המשתמש לאפליקציה בוצעה בהצלחה.	Backgrounds	register_user_main.jpg

רקע הפתיחה עם הודע שההתחברות של המשתמש לאפליקציה בוצעה בהצלחה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ

Client\Design\Backgrounds\ wait.jpg	עיצוב גרפי של האפליקציה.	רקע חלון המודיע למשתמש להמתין.	Backgrounds	wait.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ Error Detection Faces\ no_face.jpg	גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	רקע והקפצתהודעת שגיאה שהמשתמש הכניס תמונה ללא פנים.	Error Detection Faces	no_face.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה ללא פנים.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Error Detection Face\ to_many_faces.jpg	גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	רקע והקפצתהודעת שגיאה שהמשתמש תמונה עם יותר מפרצוף אחד.	Error Detection Faces	to_many_faces.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה עם יותר מפרצוף אחד.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Error Detection Face\ no_gallery.jpg	גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס גלריה כלל.	Error Detection Faces	no_gallery.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס גלריה כלל.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ wrong_username_ or_password.jpg	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.	Start Errors	wrong_username_or_ password.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ no_password.jpg	גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצתהודעת שגיאה שהמשתמש לא הכניס סיסמא בעת הרשמה לאפליקציה.	Start Errors	no_password.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס סיסמא בעת הרשמה לאפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ no_username.jpg	גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש בעת הרשמה לאפליקציה.	Start Errors	no_username.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש בעת הרשמה לאפליקציה

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ password_short.jpg	גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצתהודעת שגיאה שהמשתמש הכניס סיסמא קצרה מדי בעת הרשמה לאפליקציה.	Start Errors	password_short.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס סיסמא קצרה מדי בעת הרשמה לאפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ username_already_in_use.jpg	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצת הודעת שגיאה שהמשתמשהכניס שם משתמש שכבר תפוס בעת הרשמה לאפליקציה.	Start Errors	username_already_in_use.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס שם משתמש שכבר תפוס בעת הרשמה לאפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\ username_short.jpg	גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש קצרמידי בעת הרשמה לאפליקציה.	Start Errors	username_short.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש קצר מידי בעת הרשמה לאפליקציה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Documentation\ Project_File.docx	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	להסביר הרחבה על כל מה שקשור לפרויקט הנ"ל.	Documentation	Project_File.docx

להסביר הרחבה על כל מה שקשור לפרויקט הנ"ל.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Documentation\ Specification_Doc.docx	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	הצעת הרעיון של הפרויקט הנ"ל.	Documentation	Idea_Suggestion_Doc.docx

הצעת הרעיון של הפרויקט הנ"ל.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ

Documentation\ Specification_Doc.docx	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	אלגוריתמיקה ראשונית של הפרויקט הנ"ל.	Documentation	Specification_Doc.docx

אלגוריתמיקה ראשונית של הפרויקט הנ"ל.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Communication_Settings.py	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	קובץ כפול בצד השרת והלקוח שנועד להבטיח שהלקוח יוכל לתקשר עם הלקוח.	General Server	Communication_Settings.py

ההגדרות הכפולות של התקשרות. **משתנים:**

```
Host = '127.0.0.1'  #
Port = 9998 # the port. important that the clients
and the server be in the same port
Max_Data_Size = 9 # the biggest size of data that
can be sent. 999,999,999 bits.
```

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Run_Server.py	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	תמידית של השרת, נתינת נתונים נכונים ותקינים לשרת, דאוג לתמידות נכונה ותקינה של השרת.	General Server	Run_Server.py

יוצר קובץ DB אם לא קיים, אם קיים רק פותח אותו. מאפס את האקטיביות של כל המשתמשים, ליתר ביטחון, יוצר שרת חדש בלולאה מתמדת.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Server_Functions.py	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	פונקציות סנכרון\הזרת\קבלת מידע מלקוחות, יחד עם פונקציות אלגוריתמיות לשמירת תמונות והיסטוריה.	General Server	Server_Functions.py

אלמוג גל. Face Founder.

פונקציית יצירת שרת, יוצרת שרת עם כתובת מסויימת שמוסכת על הלקוחות, ומלה לקלוט
עד 5 לקוחות בו זמנית)listen(, וקוראת ליישר המפתח הפרטי של השרת באמצעות RSA.
בסוים כל התהליך, קם שרת מרובה לקוחות.

פונקציית השרת האקטיבי שמורצת בלולאה אינסופית קולטת לקוחות חדשים, מוצאים לקוחות בעייתיים, ומפרידה בין לקוחות שמחכים לקבלת הודעה משרת.
השרת נכתב
באמצעות select. אם הלקוח הוא עדיין socket, סימן שעוד לא בוצע חיבור ולכן יש לחבר בין השרת ללקוח. בחיבור השני הוא כבר לא socket
ולקוח לכל דבר. בכל מקרה השרת מחכה
להודעה מהלקוח. כל הודעה עוברת לפונקציה שליחת הודעה בהתאם לבקשת הלקוח.

פונקציית שליחת הודעות בהתאם לבקשת הלקוח מחכה להודעה מהלקוח. ברגע שמגיעה הודעה היא פועלת בהתאם. אם זהו הודעת בקשה: יציאה — ניתוק
הלקוח באופן נקי: להפוך
אותו ללא אקטיבי ולשמור את המידע שלו ומ)ראה ארכיטקטורת רשת, עמוד 11(.

פונקציית יצירת מפתח סימטרי, שיוצרת על פי אלגוריתם הRSA מפתח סימטרי פרטי, יחד
עם השרת. **משתנים:** min value — ערך מינימלי,max value — ערך מקסימלי.

הוספת אפסים- מוסיפה אפסים לגודל ההודעה, כך שהשרת ידע מה גודל ההודעה, בנוסף
מודיעה ללקוח אם ההודעה גדולה מידי. **משתנים:** str1 — איזושהי מחרוזת.

טיפול הודעות דואגת שהודעות גדולות מדי)מידע של קובץ תמונה(לא יודפסו אלה יודפס
במקום מידע של קובץ תמונה. **משתנים:** message — ההודעה.

פונקציית למצוא רדאר שמדפיסה את נתוני הלקוח ממנו התקבלה ההודעה. **משתנים:** לקוח. פונקציית יצירת גלריה יוצרת בpath של המשתמש הנוכחי של

האדם הנוכחי path של
גלריית תמונות שתהיה מוכנה להעלאת תמונות. שולחת ללקוח שהגלריה נוצרה בהצלחה.
משתנים: שם משתמש)כדי להוציא את הpath שמיות נתונים של המשתמש(, שם הגלריה,
מבנה הנתונים.

מחיקת מידע לא נחוץ, עוברת על כל המידע שברשות משתמש מסוים ובודקת האם שלושת התיקיות של אדם, גלריה ותמונות נמצאת קיים. אם לא השרת מוחק
תיקייה זו. **משתנים:** שם
המשתמש, מבנה הנתונים.

יצירת תיקייה מקבלת שם של תיקייה ויוצרת תיקיית תהליך חדשה בpath המידע שלהמשתמש. בנוסף גם מקבלת תמונה של אדם אותו
מחפשים. בודקת ששם התיקייה לא
נמצא שוב ואם כן מוסיפה את הספרה 1 לסוים, ואחר כך 2 וכך הלאה. שולחת הודעה לשרת
שהתיקייה נוצרה בהצלחה. **משתנים:** שם משתמש, התמונה של האדם, שם של התיקייה,
מבנה הנתונים.

בדיקת משתמש חדש בודקת האם שם המשתמש והסיסמא תקניים: מעל 4 תווים כל אחד, שם המשתמש לא קיים כבר, הוקלד איזשהו שם משתמש
ואיזשהי סיסמא. במידה ושנה טעות השרת מודיעה על הטעות ללקוח. אם הכל תקני השרת מודיע ללקוח שהחשבון נוצר בהצלחה. **משתנים:** שם
משתמש אתו רוצים לפתוח את החשבון, סיסמא של שם
המשתמש, מבנה נתונים.

בדיקת משתמש קיים בודקת האם שם המשתמש תואם את הסיסמא. אם לא שולח הודעה ללקוח ששם המשתמש או הסיסמא אינם נכונים. בנוסף
בודק שהמשתמש לא אקטיבי, ושאין ניסיון חיבור לאותו משתמש במקביל. אם יש כזה ניסיון שולח הודעה שלא ניתן להתחבר
לאותו משתמש במקביל. אם הכל תקין שולח הודעה שהכניסה לחשבון בוצעה בהצלחה.

אלמוג גל. Face Founder.

משתנים: שם משתמש אתו רוצים להיכנס את החשבון, סיסמא של שם המשתמש, מבנה נתונים.

הכנסת אדם לגלריה מקבל אדם, ומכניסה אותו לגלריה שנמצאת בתיקייה הכי עכשווית. מחזירה הודעה שהאדם הכנס בהצלחה. **משתנים:** שם משתמש, התמונה, השם של
התמונה, מבנה נתונים.

מציאת האדם שולחת למשתמש את התמונות בהם האדם בתיקייה הכי עכשווית המבוקש נמצא בגלריה של אותה תיקייה. הוא נותן את הpaths של הגלריה והאדם לפונקציית מציאת אדם שמחזיר לו רשימה של התמונות בהם נמצא האדם המבוקש. מתוך הרשימה הזו
השרת שולח תמונה כל פעם ללקוח, ומחכה מעט זמן כדי שהלקוח יעשה תהליך בצד שלו.
כאשר הרשימה נגמרת שולח השרת שנגמרו כל התמונות בהם האדם נמצא לשרת.
משתנים: הלקוח, שם המשתמש, מבנה הנתונים.

העברת שמות תיקיות . השרת יקבל את הpath של הלקוח גם בעזרת פרוטוקול הsql שברשותו. כך ידע לאן לגשת ולהוציא את כל תיקיות ההיסטוריה שקיימות לאותו משתמש במחשב לו. תיקיות אלו מעולות שר לשרת, אך לא ישר ללקוח. בעזרת חיסור הרשימות ידע השרת איזו תיקיות עוד לא העלה ללקוח ואת שמם ישלח ללקוח עם הוראה ליצור תיקיות
אלו בpath ההיסטוריה של הלקוח. **משתנים:** לקוח, שם משתמש, מבנה נתונים.

שליחת קבצי היסטוריה שלחת ללקוח את כל קבצי ההיסטוריה בתיקיות ההיסטוריה החדשות, כדי ששמור אותם אצלם. הוא שולח הודעה בה שם התיקייה, התת-תיקייה, שם
הקובץ, מידע בנארי של הקובץ. בסוף שולח השרת ללקוח שאין יותר קבצים. **משתנים:**
הלקוח, שם המשתמש, התיקיות.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Database.py	העברת נתונים למבנה הנתונים בהתאם לבקשת השרת, באופן תקין, נכון ומסודר.	הכנסה, הוצאה וכל טיפול שהוא במבנה הנתונים הקיים, לצורך תפקוד תקין של השרת.	Database	Database.py

פעולות בנוגע לטבלת הsql כגון להפוך משתמש לאקטיבי, ללא אקטיבי, לתת למשתמש path היסטוריה במחשב של השרת ולשמור אחת כזו,
יצירת DB חדש\לפתוח את הקובץ
הקיים ולפעול בו, לקחת מידע ברשימות מן עמודות, להכנס שם משתמש עם סיסמא, למחוק משתמש, להשיג את path ההיסטוריה של המשתמש, להכניס את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, להכניס את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, להחזיר את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, למחוק את הטבלה, להחזיר את כל הנתונים בטבלה, מחזירה true אם הטבלה מורצת מcmd או מpython. **משתנים:** history paths – איפה לפתוח תיקות היסטוריה חדשות, Server path – איפה השרת מורץ וpath database – איפה מבנה הנתונים הקיים\ איפה לפתוח אחד
חדש.

אלמוג גל. Face Founder.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Detect_faces.py	לאתר ולהשוות פנים מבין תמונות שונות.	אלגוריתמיקה של מציאת והשוואה של פרצופים בין תמונות שונות, תוך כדי שימוש במכונה לומדת מאומנת.	Detection and Compression Faces	Detect_faces.py

פונקציית האם פנים מקבלת תמונה אשר אותה מכניסה למכונה שמכבר מזהה מאזכרים של פנים כמו: עיינים, אף, פה, תווי פנים וכדומה. המכונה עוברת על כל חלקי התמונה, סורקת, וכן יודעת כמה פרצופים יש בתמונה. המכונה עושה זאת משום שכבר אומנה על מיליוני תמונות של פרצופים ומלה לזהות במהירות ובדיוק רב כמה פרצופים יש בתמונה. המכונה המאומנת מחזירה שקר יחד עם הודעה "יותר מידי פרצופים" כאשר יש יותר מפרצוף אחד בתמונה, מחזירה שקר יחד עם הודעה "אין פרצופים" כאשר אין פרצופים בתמונה. המכונה מחזירה אמת ושהכל תקין אם יש פרצוף אחד בתמונה. זה לחלק אחד בתוכנית כאשר המשתמש מכניס תמונה של לקלט של אדם, ועל השרת לבדוק האם יש פרצוף אחד בתמונה או לא. **משתנים:** קובץ תמונה.

החלק השני קשור להשוואה בין תמונה לגלרייית תמונות והשאלה היא האם האדם בתמונה הזו הוא אותו אדם בתמונה השנייה. לשם כך יש את פונקציית השוה פנים שמקבלת תמונה של אדם וגלריה. הפונקציה בפועל בודקת כל פעם בין שתי תמונות, בין האדם המבוקש לתמונה אחת מהגלריה עד שהא עוברת על כל תמונות הגלריה. גם פה מדובר במכונה
לומדת, שקיבלת מיליוני tupels וlists של אנשים זהים עד שידעה לייצר את מספר הדמיון בין שתי תמונות. כמובן, ככל שהתמונה יותר ברורה, חדה ובהירה אחוזי הדמיון עולים. אני מריץ כמה תמונות במקביל על מנת לייעל את זמן העבודה, מספר התמונות שמורצות ב זמנית שוה למספר הCPUs שיש לך על המחשב)לי יש 4(. המכונה שומרת את התמונות בהם יש
אחוזי דמיון גבוהים, וגם פה נתתי למכונה לא רק אחוז דמיון גבוה כדי שלא יתפספס תמונות נכונות. המכונה מחזירה את התמונות בהם האדם נמצא. **משתנים:** תיקיית תמונות – גלריה, תמונה של אדם, מספר CPUs)מוגדר אצלי כ4(, רף דמיון מינימלי)מוגדר אצלי כ0.6(, הראה
דמיון בין אנשים)בולאני(.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Encryption_And_Decryption.py	לדאוג שמידע אישי ורגיש של הלקוח לא יחשף בעת תקשורת עם השרת.	קובץ כפול הדאג להצפין. ולפענח הודעות לפי מפתח ואלגוריתם זהה בני הצדדים, וכן להבטיח אבטחה הבנה של ההודעות בתקשורת.	Security Server	Encryption_And_Decryption.py

אלמוג גל. Face Founder.

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח
והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים יודע מה המידע בהודעה. אלגוריתם <u>ההצפנה</u> בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה
)הודעה(מסוג
רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו)באמצעות פונקציית ord(. את המספר
הנ"ל הופכים)נגד 1546 הופך
ל6451(. מחסירים ממנו 100 עד שנמצא בין 0 ל255, בנתיים שנו מונה)ערכו הראשוני 40(שעולה ,בכל החסרה של המספר ההפוך,
במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר
הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה
למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג
התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה
הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח יהיה מסוגל להעביר את האותיות הascii
למקור, שכן הוא יודע כמה)בעזרת צירוף המונה ליד כל אות(פעמים להכפל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית
ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל
ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק מידע מוצפן. גם בעזרת brute force לא ימצא מהו
הפאקטה המקורית כי אין ברשותו את המפתח. כאמור הוא יכול לנחש פעם את המפתח ועלול לנסות brute force, אבל כמובן דבר זה יקח לו שנים.
חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי
מאלפי תווים אבל זוהי בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהיא רשימה, עוברת pickle, שהופך אותה לstring ארוך.
דבר שמקשה עוד יותר על גורם שלישי
עין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם
רק השרת והלקוח
יודעים את ההודעות שהם שולחים זה לזה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Preparation_For_Encyption.py	לדאוג שמידע אישי ורגיש של הלקוח לא יחשף בעת תקשורת עם השרת.	דואג ליצירת מספרים ומפתחות ציבוריים ופרטיים סימטריים עם הלקוח, דבר שדואג להצפנה ופיענוח גם, תקשורת תקינה ומאובטחת.	Security Server	Preparation_For_Encyption.py

אלמוג גל. Face Founder.

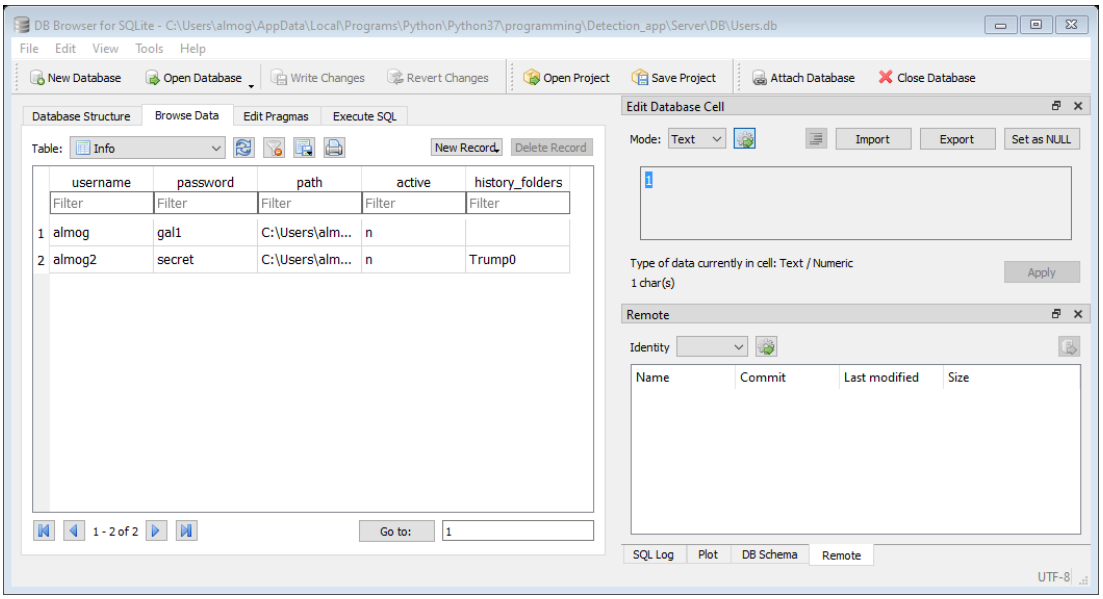
RSA: כאשר השרת נהיה אקטיב, הוא קורא לפונקציה שיוצרת את המפתח הציבורי.
המפתח הציבורי נשמר כמשתנה גלובלי לאורך כל זמן הרצת השרת.

המפתח הציבורי בנוי משני מספרים: מספר ראשון- n- מכפלת שני מספרים ראשונים רנדומליים. m- ביצוע פונקציית אולר)Euler(על המספר הראשון. מספר שני – לאחר יצירת m התחלה של חיפוש מספר החל מ5)כל מספר ראשוני קטן עובד, בחרתי ב5(של מספר שגם לא זוגי וגם זר לחלוטין)אין להם מחלק משותף(מציאת המספר הנ"ל הינו המספר השני - e. הפתח הציבורי, כשמו הוא, נשלח לכל לקוח שמעוניין בו. לאחר מכן השרת יוצר מפתח פרטי שרק הוא יודע, זאת באמצעות אלגוריתם פשוט לשרת, אך לניחוש פורץ מאוד מסובך. אני בחרתי ליצור מספר k שיהיה שווה לאחד וגדיל את עצמו באחד בלולאה. בכל פעם בלולאה אני קורא לפונקציה בה אני מכפיל את k בm מוסיף להם אחד ומחלק בe. אם התוצאה שלמה הא המספר הפרטי של השרת – d.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה הא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות
בינהן כך שרק הם יודעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה)שאפרט בהמשך(ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין
ברשת, בלי המספר הפרטי של השרת d הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של הלקוח והשרת.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\DB\ Users.db	שמירת נתוניהלקוחות באופן מסודר ותמיד אך ניתן לשינוים.	שמירת נתונים במסד נתונים כלשהו, באופן תמידי, נוח המאפשר שינוים.	Database	Users.db



1. בעמודה שם משתמש יהיה שם המשתמש של המשתמש. שם המשתמש תקין וייחודי. באמצעות sql ניתן לקבוע שעמודה זו תהיה ייחודית. משתנים מסוג string(text בפייתון).

2. בסיסמא תהיינה סיסמת המשתמש. סיסמת המשתמש תהיה תקינה ותמיד תהיה קשורה לשם המשתמש תחת נכנסה. משתנים מסוג text.

3. במיקום יהיה מיקום שמירת תמונות המשתמש אצל השרת. כך השרת ידע לאיפה לגשת כאשר הוא צריך מידע הקשור ללקוח, למשל על מנת לעלות את היסטוריית הלקוח יש לדעת איפה היא נמצאת- במיקום של, וכך ניתן לגשת אליה. משתנה מסוג text.

4. במחובר יהיה כן או לא לפי האם המשתמש מחובר או לא. כאשר ישנה התנתקות מכל סוג שהיא השרת שם לב שהלקוח מכתובת מסיימת לא מחובר\לא מגיב וכך השרת ידע שהלקוח התנתק מאיזשהי סיבה. כך בעצם מעדכן השרת את העמודה. כך גם ניתן לוודא שאף לקוח לא מתחבר למשתמש שלו במקביל ולמנוע את קריסת השרת. משתנה מסוג text.

5. בתיקיות היסטוריה יהיה תיקיות ההיסטוריה שהשרת כן העלה ללקוח. כך השרת ידע אילו תיקיות חדשות נוספו מהפעם הקודמת שהוא העלה היסטוריה כי בכל העלאת היסטוריה השרת מעדכן בהתאם את העמודה הנ"ל. כך בעצם מובטח יעילות מעל הבחינות כשאר השרת ידע אילו תיקיות היסטוריה עליו לעלות. משתנה מסוג text.

בסיס הנתונים:

בפרויקט יש בס"ס נתונים הנכתב באמצעות sql בפייתון. בס"ס נתונים זה ישתמש בסוג sqlight3. בס"ס נתונים הפנימי יהיה טבלה, בעלת עמודות ושורות, בה ישמור את הנתונים באופן מאובטח ומסודר. בטבלה חמש עמודות: שם משתמש, סיסמא, מיקום מחובר, תיקיות היסטוריה.

1. בעמודה שם משתמש יהיה שם המשתמש של המשתמש. שם משתמש תקין ויחודי.
באמצעות sql ניתן לקבוע שעמודה זו תהיה יחודית. משתנים מסוג string(text בפייתון).

2. בסיסמא תהיינה סיסמת המשתמש. סיסמת המשתמש תהיה תקינה ותמיד תהיה קשורה לשם המשתמש תחתיו נכנסה. משתנים מסוג text.

3. במיקום יהיה מיקום שמירת תמונות המשתמש אצל השרת. כך השרת ידע לאיפה לגשת כאשר הוא צריך מידע הקשור ללקוח, למשל על מנת לעלות את היסטוריית הלקוח יש לדעת
איפה היא נמצאת- במיקום של, וכך ניתן לגשת אליה. משתנה מסוג text.

4. במחובר יהיה כן או לא לפי האם המשתמש מחובר או לא. כאשר ישנה התנתקות מכל סוג שהא השרת שם לב שהלקוח מכתובת מסויימת לא מחובר\לא מגיב וכך השרת ידע
שהלקוח התנתק מאיזשהי סיבה. כך בעצם מעדכן השרת את העמודה. כך גם ניתן לוודא שאף לקוח לא מתחבר למשתמש שלו במקביל ולמנוע את קריסת השרת. משתנה מסוג
text.

5. בתיקות היסטוריה יהיה תיקות ההיסטוריה שהשרת כן העלה ללקוח. כך השרת ידע אילו תיקות חדשות נוספו מהפעם הקודמת שהוא העלה היסטוריה כי בכל העלאת
היסטוריה השרת מעדכן בהתאם את העמודה הנ"ל. כך בעצם מובטח יעילות מעל הבחינות
כשאר השרת ידע אילו תיקות היסטוריה עליו לעלות. משתנה מסוג text.

כל שורה תהייה פיסת מידע - text. הסיבה מאוחרי לשמור את המידע בפיסות מידע הוא כי אין מערכים בsqlight3 ולכן אי אפשר לעשות מערך של התמונות של המשתמש)רק לעלות תמונה אחת כל פעם — מאוד לא יעיל(. ובשביל לפתור בעיה זו בדרך היעלה ביותר נאלצתי לפתוח במחשב השרת תיקייה לכל שם משתמש, התיקרא על שמו הייחודית ושם לשמור את
התמונות. בשביל לא להגיע למצב שתמונה תישמר עם אותו השם פעמיים אני אוסיף את
התאריך והשעה המדויקת של שמירת התמונה על שרת המחשב.

תיקיות היסטוריה	מחובר	מיקום	סיסמא	שם משתמש
Yossi, Miriam, Dana	n	Db\Almog	1234	Almog
	y	Db\Dani1	Hello123	Dani1

לדוגמא אלמוג נכנס ל"היסטוריה אישית". התוכנה תחפש במיקום Db\Almog את כל התיקיות של האנשים שאלמוג רצה לחפש. תחת כל תיקייה יהיו האנשים שנמצאו להיות מתאימים לשם התיקייה הגלריה של האנשים והתמונה המקורית של אותו אדם. דוגמא
נוספת היא כאשר השרת יפול כל המשתמשים יהפכו למנותקים וכך לא יגעו משתמשים. דוגמא נוספת היא שלא תהיינה תיקייה עם אותו שם בשל העובדה שהתיקיות הם לפי שמות המשתמש שהם דבר ייחודי לפי הdb, וגם התמונות ישמרו תחת שם התמונה ותאריך ושעה
של תמונה.

אלמוג גל. Face Founder.

ביבליוגרפיה

Websites:

Google

Stack Overflow

Wikipedia

סיכום אישי \ רפלקציה

עבור העבודה על הפרויקט הייתה מעשירה ומעצימה מבחינת ידע תכנותי מודרני)למשל
כתיבת קוד טובה בפייתון(. התחלתי את הפרויקט עם לא הרבה ידע לגבי שרת-לקוח, רשת, שפת פייתון וGUI. אם כן, בסוף התהליך אני יודע טוב את כולם.
אז במבחן התוצאה
הפרויקט העשיר רבות את הידע שלי במחשבים והרחיב לי אופקים. אם כן, מנגד, הפרויקט לדעתי היה מאוד ארוך וגם קשה. הוא חלקים ואלגוריתמים שלקחו לי
שבועות, דבר שכמעט שבר אותי כמה פעמים במהלך הפרויקט. בכל זאת, המשכתי. כנראה שכן היה בי הרצון
לעשות את הפרויקט, לנצח אתגר הנראה בעיני בלתי אפשרי, וכמובן לקבל עוד 5 יח"ל. השקפתי לגבי העולם הטכנולוגי שונתה לחלוטין, התחלתי להבין לעומק
איך דברים עובדים ומה ההיגיון מאחורי מוצרים טכנולוגיים רבים בשוק. הרחבת הידע לאט לאט הביאה לי ביטחון בעולם הזה, דבר שגרם לעלייה פרבולית בקצב
ההתקדמות והידע שלי. כך אחרי מספר חודשים אני כאן כותב את הסיכום הזה, שהפרויקט שלי ברובו מוכן. אני לוקח איתי להמשך הדרך את כל הידע שצברתי
שיעזור לי גם אחרי בית ספר, אם זה בקבלת תפקיד טכנולוגי טוב בצבא שקיבלתי הוא אם זה אח"כ באזרחות. אני מודה, הוא לי קשים שלא
חשבתי שאתגבר עליהם וכמה לילות חסרי מנוחה בגלל החשיבה הלא מפסקת על עקיפת הבעיות. אך עדיין כל שבוע ישבתי ונסיתי לעקוף, לפתור איכשהו
ואפילו לצמצם בעיות כלשהם. דוגמא לקושי שהיה לי הוא בהקמת השרת הראשוני, בשנה הקודמת לתחילת
הפרויקט)כיתה י'(לא צברתי ידע מספק לתחילת הפרויקט. לא הצלחתי ולא הבנתי לקוח- שרת, כתיבת הקוד שלי הייתה לוקה בחסר וזה דבר שמאוד הקשה
עליי בכתיבה של הבסיס הראשוני של הפרויקט שלי. כחודש ישבתי רק באימון על כתיבת קוד של שרת-לקוח. ואז
שדרגתי למודל של שרת-לקוחות וכן הלאה. ללא ספק הייתה לי עלייה גדולה מתחתית מסיימת להצלחה יפה מאוד לטעמי. הוא קשיים מיאשים, החלטתי
לוותר על רוב הלמדת מכונה בפרויקט שלי לאחר שבועיים של ניסיונות, וזה הציף אותי בשאלות כמו אם אתה לא
מצליח לבצע את המרכז בפרויקט שלך איך תעשה את שאר הפרויקט או האם הפרויקט שלי יהיה לא טוב. אבל המשכתי בשל, וטוב שכך. אני מסתכל אחורנית,
ואני מבין שאני יכול
לעשות וללמוד המון דברים, מסבכים קשים ומיאשים ככל שיהיו, אם רק יש לי את הרצון. הימלות בי קיימת. זה טוב לאתגר את עצמך כדי להגיע למסקנה כזו,
מסקנה שנותנת לך ביטחון להמשך. אם כי ימלתי לבצע דברים בצורה הרבה יותר נכונה, ושגיתי. עשיתי טעות ש"רצתי" ישר לכתיבת הפרויקט עם שיטה מסיימת
במקום לשבת ולקרוא יום ויומיים על האופציות האחרות. זה בא לידי ביטוי באופן הכי מובהק בבחירת יצירת מסך GUI עם
tkinter. זה היה שיטה נוראית, מסורבלת וקשה שהעסיקה אותי חודשים בבניית הGUI. הוא שיטות פי כמה וכמה יותר קלות, טובות ומהירות ובכל זאת
מהרתי. אבל עדיין טעיות כאלה הוא בשבילי גם אתגר. כן אם הייתי חושב יותר ופחות כותב קוד הייתי מגיע לתוצאה יותר
טובה. אבל עדיין, זה פעם ראשונה שאני בונה אפליקציה משלי וזה לגיטימי שאני אטעה.
לסיכום, הפרויקט היה קשה, אך מעצים ואני מרוצה מהתוצאה ובחב הדרך. תלמיד סייבר,
כיתה יב', אלמוג גל.